

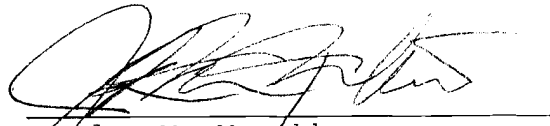
AMENDMENT

SERIAL NO: 09/429,357

Page: 26

condition for allowance. Accordingly, reconsideration of the present application is respectfully requested.

Respectfully submitted,

A handwritten signature in black ink, appearing to read 'Jordan M. Meschkow', is written over a horizontal line.

Jordan M. Meschkow  
Attorney for Applicants  
Reg. No. 31,043

Dated: 8 January 2003

Jordan M. Meschkow  
Meschkow & Gresham, P.L.C.  
5727 North Seventh Street  
Suite 409  
Phoenix, AZ 85014  
(602) 274-6996

AMENDMENT

SERIAL NO: 09/429,357

Page: 27

## APPENDIX A

A marked-up copy of Figure 5 showing the modifications is presented on the following page.

29/421,341



approved  
by  
Examiner  
GCE

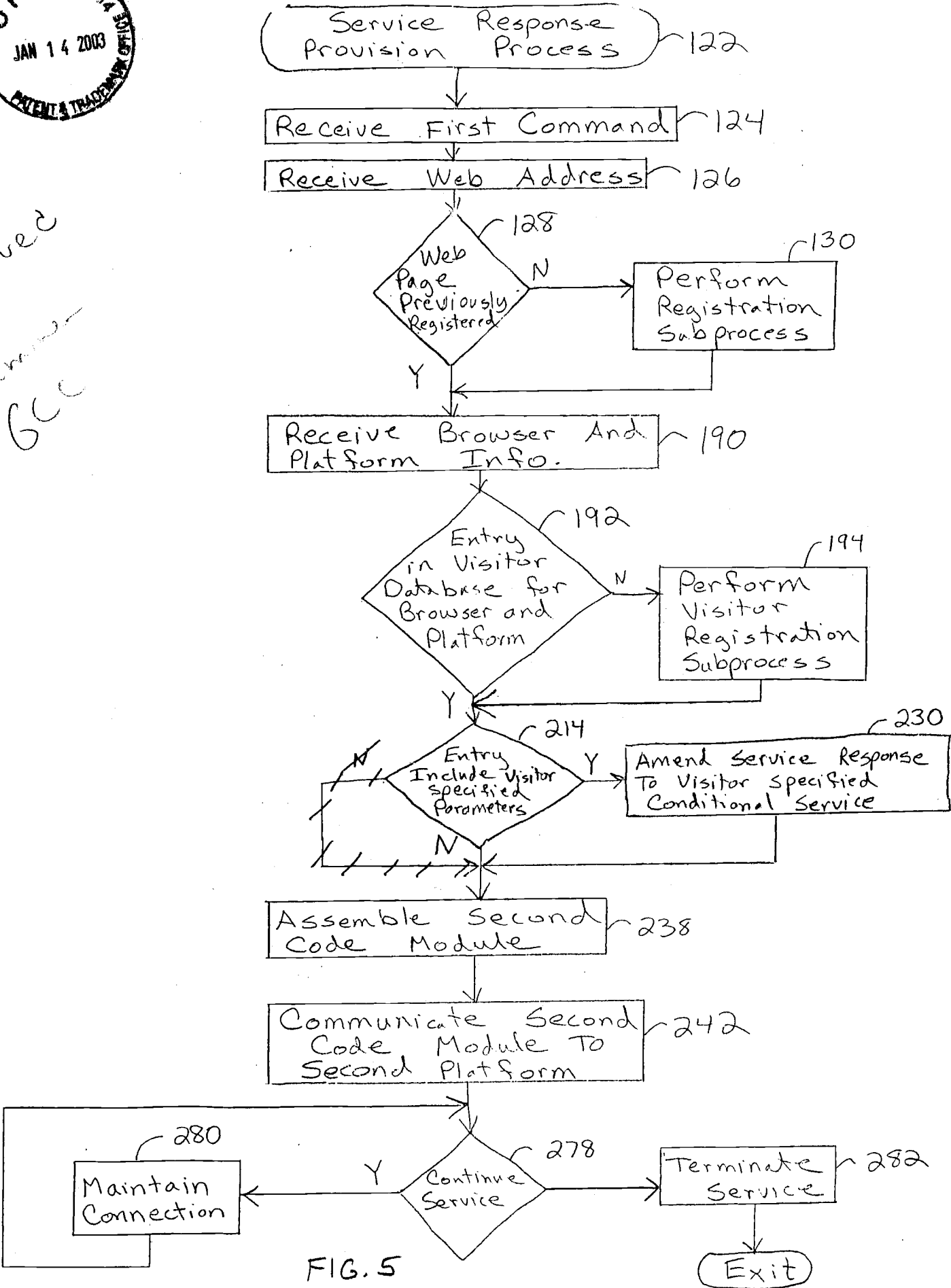


FIG. 5

AMENDMENT

SERIAL NO: 09/429,357

Page: 29

## APPENDIX B

**Marked-up copy of the paragraph on page 4, beginning at line 8:**

The above and other advantages of the present invention are carried out in another form by a computer readable code module for adding function to a Web page. The code module is configured to be embedded in the Web page which is generated in a HyperText Markup Language (HTML), and is configured for automatic execution when the Web page is downloaded to a client machine supporting a graphical user interface and a Web browser. The computer readable code module includes means for communicating a Web address of the Web page to a server system via a network connection to initiate a download of a second computer readable code module to the client machine. The computer readable code module further includes means for communicating first information characterizing said Web browser to said server and means for communicating second information characterizing said client machine to said server. In addition, the computer readable code module includes means for initiating execution of said second computer readable code module following the download of the second computer readable code module and means for providing a comment tag informing the [HTML of the Web page] Web browser to ignore the initiating means.

**Marked-up copy of the paragraph on page 6, beginning at line 18:**

Web address 38 is a Universal Resource Locator (URL), or a string expression used to [located] locate Web page 34 via

AMENDMENT

SERIAL NO: 09/429,357

Page: 30

network 28. It should be readily apparent to those skilled in the art that first processor platform 22 also includes additional components such as input/output lines, a keyboard and/or mouse, and a display terminal which are not shown for the sake of clarity. In addition, memory 32 also contains additional information, such as application programs, operating systems, data, etc., which also are not shown for the sake of clarity.

**Marked-up copy of the paragraph on page 9, beginning at line 15:**

A third command line (LINE NO. 3) 100 starts a new script. Third command line 100 also contains a comment tag 102 used to allow [the HTML code of Web page 34] Web browser 52 to ignore a fourth command line (LINE NO. 4) 104. Fourth command line 104 contains a second command 106 that initiates execution of second code module 90 that was downloaded to temporary memory 54 of second processor platform 24. A fifth command line 108 terminates comment tag 102 and terminates the script begun on third command line 100.

**Marked-up copy of the paragraph on page 27, beginning at line 20:**

In summary, the present invention teaches of a method and system for adding function, such as streaming media or other media services to a Web page, through the implementation of a simple code module embedded in the HTML of the Web page. The code module is compatible with Web browsers which adhere to the standards for HyperText Transfer Protocol (HTTP) because it is implemented using a common subset of the current HTML standard command set. In addition, the code module is easily distributed through the Internet, and is readily copied and pasted into a Web

## AMENDMENT

SERIAL NO: 09/429,357

Page: 31

page during Web page development activities, and undergoes automatic execution and registration with minimal effort by the Web page developer. The present invention is able to tailor the added function based on information about the Web page in which it is embedded and based on visitor specified preferences.

**Marked-up copy of the Abstract:**

A computer network (20) includes a first processor (22) for maintaining a Web page (34) having [a] an embedded first code module (36) [embedded therein and being] and accessible through a Web address (38). A second processor (24) [is in communication with the first processor (22) via the Internet (28). The second processor (24)] supports a Web browser (52) for downloading the Web page (34) and executing the first code module (36). When executed, the first code module (36) issues a first command (93) to retrieve a second code module (90) [. A] from a server system (26) [in communication with the second processor (24) receives the first command (93)]. The server system (26) includes a database (68) having [stored therein] a service response (162, 176, 186) [in association] associated with the Web address (38). A processor (62) [in communication with the database (68)] assembles the second code module (90) having the service response (162, 176, 186). [Upon retrieving] When the second code module [at the second platform (24)] is retrieved, the first code module (36) issues a second command (106) to initiate execution of the second code module (90) [at the second processor (24) in order] to provide added function to the Web page (34) [, such as streaming audio or video media, banners, informational feeds, and so forth].

AMENDMENT

SERIAL NO: 09/429,357

Page: 32

## APPENDIX C

**Marked-up copy of the claims 1-3, 6, 7, 11, and 20 showing the modifications:**

1. (Amended) A method of operating a computer network to add function to a Web page comprising the steps of:

downloading said Web page at a processor platform, said downloading step being performed by a Web browser;

when said Web page is downloaded, automatically executing a first code module embedded in said Web page;

said first code module issuing a first command to retrieve a second code module, via a network connection, from a server system;

receiving, at said server system, first information characterizing said Web browser in response to said executing step;

receiving, at said server system, second information characterizing said processor platform in response to said executing step;

storing said first and said second information in a visitor database of said server system, said first and said second information being associated with a tracking index; and

said first code module issuing a second command to initiate execution of said second code module at said processor platform, said second code module being responsive to said first and second information.

2. (Amended) A method as claimed in claim 1 wherein said [downloading step is performed by a] Web browser [employing] employs HyperText Transfer Protocol (HTTP) and said first code module is generated in a HyperText Markup Language (HTML).

AMENDMENT

SERIAL NO: 09/429,357

Page: 33

3. (Amended) A method as claimed in claim 2 wherein said Web page is generated in said HTML, and said first code module includes a comment tag informing said [HTML of said Web page] Web browser to ignore said second command.

6. (Amended) A method as claimed in claim 5 further comprising the steps of:

forming a service response related to said profile of said Web page;

storing said service response in association with said Web address; and

accessing said service response when said first code module issues said first command so that said service response is included in said second code module.

7. (Amended) A method as claimed in claim 1 [wherein said downloading step is performed by a Web browser, and said method comprises] further comprising the steps of:

[receiving, at said server system, first information characterizing said Web browser in response to said first command;

receiving, at said server system, second information characterizing said processor platform in response to said first command;]

assembling, at said server system, said second code module having a service response[, said second code module being responsive to said first and said second information];

transferring said second code module to said processor platform; and

executing, at said processor platform, said second code module to indicate said service response.



AMENDMENT

SERIAL NO: 09/429,357

Page: 34

11. (Amended) A method as claimed in claim 1 [wherein said downloading step is performed by a Web browser, and said method further comprises] further comprising the steps of:

applying said tracking index to said processor platform in response to said first and second information; and

using said tracking index at said server system to track and identify said processor platform.

20. (Amended) A computer readable code module for adding function to a Web page, said code module configured to be embedded in said Web page generated in a HyperText Markup Language (HTML) and configured for automatic execution when said Web page is downloaded to a client machine supporting a graphical user interface and a Web browser, said computer readable code module including:

means for communicating a Web address of said Web page to a server system via a network connection to initiate a download of a second computer readable code module to said client machine;

means for communicating first information characterizing said Web browser to said server system;

means for communicating second information characterizing said client machine to said server system;

means for initiating execution of said second computer readable code module following said download of said second computer readable code module, said second computer readable code module being responsive to said first and second information; and

means for providing a comment tag informing said [HTML of said Web page] Web browser to ignore said initiating means.

# 6/B

<b>Notice of Allowability</b>	<b>Application No.</b>	<b>Applicant(s)</b>	
	09/429,357	MCCOLLUM ET AL.	
	<b>Examiner</b>	<b>Art Unit</b>	
	Gregory L. Clinton	2154	

**-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address--**

All claims being allowable, PROSECUTION ON THE MERITS IS (OR REMAINS) CLOSED in this application. If not included herewith (or previously mailed), a Notice of Allowance (PTOL-85) or other appropriate communication will be mailed in due course. **THIS NOTICE OF ALLOWABILITY IS NOT A GRANT OF PATENT RIGHTS.** This application is subject to withdrawal from issue at the initiative of the Office or upon petition by the applicant. See 37 CFR 1.313 and MPEP 1308.

1. ☒ This communication is responsive to 1/8/03.
2. ☒ The allowed claim(s) is/are 1-6, 8-29.
3. ☒ The drawings filed on 08 January 2003 are accepted by the Examiner.
4. ☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).
  - a) ☐ All    b) ☐ Some\*    c) ☐ None    of the:
    1. ☐ Certified copies of the priority documents have been received.
    2. ☐ Certified copies of the priority documents have been received in Application No. \_\_\_\_\_.
    3. ☐ Copies of the certified copies of the priority documents have been received in this national stage application from the International Bureau (PCT Rule 17.2(a)).

\* Certified copies not received: \_\_\_\_\_.

5. ☐ Acknowledgment is made of a claim for domestic priority under 35 U.S.C. § 119(e) (to a provisional application).
  - (a) ☐ The translation of the foreign language provisional application has been received.
6. ☐ Acknowledgment is made of a claim for domestic priority under 35 U.S.C. §§ 120 and/or 121.

Applicant has THREE MONTHS FROM THE "MAILING DATE" of this communication to file a reply complying with the requirements noted below. Failure to timely comply will result in ABANDONMENT of this application. **THIS THREE-MONTH PERIOD IS NOT EXTENDABLE.**

7. ☐ A SUBSTITUTE OATH OR DECLARATION must be submitted. Note the attached EXAMINER'S AMENDMENT or NOTICE OF INFORMAL PATENT APPLICATION (PTO-152) which gives reason(s) why the oath or declaration is deficient.
8. ☒ CORRECTED DRAWINGS must be submitted.
  - (a) ☒ including changes required by the Notice of Draftsperson's Patent Drawing Review (PTO-948) attached
    - 1) ☐ hereto or 2) ☒ to Paper No. 3.
  - (b) ☐ including changes required by the proposed drawing correction filed \_\_\_\_\_, which has been approved by the Examiner.
  - (c) ☐ including changes required by the attached Examiner's Amendment / Comment or in the Office action of Paper No. \_\_\_\_\_.

Identifying indicia such as the application number (see 37 CFR 1.84(c)) should be written on the drawings in the top margin (not the back) of each sheet. The drawings should be filed as a separate paper with a transmittal letter addressed to the Official Draftsperson.

9. ☐ DEPOSIT OF and/or INFORMATION about the deposit of BIOLOGICAL MATERIAL must be submitted. Note the attached Examiner's comment regarding REQUIREMENT FOR THE DEPOSIT OF BIOLOGICAL MATERIAL.

**Attachment(s)**

- |  |  |
|--|--|
| 1 <input type="checkbox"/> Notice of References Cited (PTO-892)  | 2 <input type="checkbox"/> Notice of Informal Patent Application (PTO-152) |
| 3 <input type="checkbox"/> Notice of Draftsperson's Patent Drawing Review (PTO-948)                    | 4 <input type="checkbox"/> Interview Summary (PTO-413), Paper No. _____    |
| 5 <input type="checkbox"/> Information Disclosure Statements (PTO-1449), Paper No. _____               | 6 <input checked="" type="checkbox"/> Examiner's Amendment/Comment         |
| 7 <input type="checkbox"/> Examiner's Comment Regarding Requirement for Deposit of Biological Material | 8 <input type="checkbox"/> Examiner's Statement of Reasons for Allowance   |
|  | 9 <input type="checkbox"/> Other   |

*Zarni Maung*  
**ZARNI MAUNG**  
**PRIMARY EXAMINER**

Application/Control Number: 09/429,357  
Art Unit: 2154

Page 2

**EXAMINER'S AMENDMENT**

1. An examiner's amendment to the record appears below. Should the changes and/or additions be unacceptable to applicant, an amendment may be filed as provided by 37 CFR 1.312. To ensure consideration of such an amendment, it MUST be submitted no later than the payment of the issue fee.

Authorization for this examiner's amendment was given in a telephone interview with Jordan Meschkow, reg. no. 31,043, on March 21, 2003.

The application has been amended as follows:

Please cancel claim 7.

In claim 1, line 18, after ":", please delete "and" and insert:

---assembling, at said server system, said second code module, said second code module containing a service response related to said Web page; said second code module being responsive to said first and second information;

downloading, in response to said first command, said code module to said processor platform; and---

In claim 1, line 20, please delete " , said second code module being responsive to said first and second information"

Application/Control Number: 09/429,357  
Art Unit: 2154

Page 3

Please amend claim 6 as follows:

6. (Twice Amended) A method as claimed in claim 5, wherein said service response is related to said profile of said Web page, further comprising the steps of:

[forming a service response related to said profile of said Web page;]

storing said service response in association with said Web address;

accessing said service response when said first code module issues said first command

[so that said service response is included in said second code module].

In claim 20, line 14, after “;”, please insert:

---means for assembling, at said server system, said second computer readable code module, said second computer readable code module containing a service response related to said Web page, said second computer readable code module being responsive to said first and second information;

means for downloading said second computer readable code module to said client machine;---

In claim 20, line 17, please delete “, said second computer readable code module being responsive to said first and second information”.

Application/Control Number: 09/429,357

Page 4


Art Unit: 2154

Any inquiry concerning this communication or earlier communications from the examiner should be directed to Gregory L. Clinton whose telephone number is 703-305-3179. The examiner can normally be reached on Monday - Thursday 8:00-5:30.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Meng-Ai T. An can be reached on 703-305-9678. The fax phone numbers for the organization where this application or proceeding is assigned are 703-746-7239 for regular communications and 703-746-7238 for After Final communications.

Any inquiry of a general nature or relating to the status of this application or proceeding should be directed to the receptionist whose telephone number is 703-305-3900.

Gregory Clinton  
March 25, 2003

  
ZARNI MAUNG  
PRIMARY EXAMINER



UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE  
United States Patent and Trademark Office  
Address: COMMISSIONER OF PATENTS AND TRADEMARKS  
Washington, D.C. 20231  
www.uspto.gov

NOTICE OF ALLOWANCE AND FEE(S) DUE

7590 03/26/2003

JORDAN M MESCHKOW  
MESCHKOW & GRESHAM PLC  
5727 NORTH SEVENTH STREET  
SUITE 409  
PHOENIX, AZ 85014

EXAMINER	
CLINTON, GREGORY L	
ART UNIT	CLASS-SUBCLASS
2154	709-218000

DATE MAILED: 03/26/2003

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
09/429,357	10/28/1999	CHARLES P. MCCOLLUM	7791-A-1	5829

TITLE OF INVENTION: METHOD AND SYSTEM FOR ADDING FUNCTION TO A WEB PAGE

APPLN. TYPE	SMALL ENTITY	ISSUE FEE	PUBLICATION FEE	TOTAL FEE(S) DUE	DATE DUE
nonprovisional	YES	\$650	\$0	\$650	06/26/2003

THE APPLICATION IDENTIFIED ABOVE HAS BEEN EXAMINED AND IS ALLOWED FOR ISSUANCE AS A PATENT. PROSECUTION ON THE MERITS IS CLOSED. THIS NOTICE OF ALLOWANCE IS NOT A GRANT OF PATENT RIGHTS. THIS APPLICATION IS SUBJECT TO WITHDRAWAL FROM ISSUE AT THE INITIATIVE OF THE OFFICE OR UPON PETITION BY THE APPLICANT. SEE 37 CFR 1.313 AND MPEP 1308.

THE ISSUE FEE AND PUBLICATION FEE (IF REQUIRED) MUST BE PAID WITHIN THREE MONTHS FROM THE MAILING DATE OF THIS NOTICE OR THIS APPLICATION SHALL BE REGARDED AS ABANDONED. THIS STATUTORY PERIOD CANNOT BE EXTENDED. SEE 35 U.S.C. 151. THE ISSUE FEE DUE INDICATED ABOVE REFLECTS A CREDIT FOR ANY PREVIOUSLY PAID ISSUE FEE APPLIED IN THIS APPLICATION. THE PTOL-85B (OR AN EQUIVALENT) MUST BE RETURNED WITHIN THIS PERIOD EVEN IF NO FEE IS DUE OR THE APPLICATION WILL BE REGARDED AS ABANDONED.

HOW TO REPLY TO THIS NOTICE:

I. Review the SMALL ENTITY status shown above.

If the SMALL ENTITY is shown as YES, verify your current SMALL ENTITY status:  
A. If the status is the same, pay the TOTAL FEE(S) DUE shown above.  
B. If the status is changed, pay the PUBLICATION FEE (if required) and twice the amount of the ISSUE FEE shown above and notify the United States Patent and Trademark Office of the change in status, or

If the SMALL ENTITY is shown as NO:  
A. Pay TOTAL FEE(S) DUE shown above, or  
B. If applicant claimed SMALL ENTITY status before, or is now claiming SMALL ENTITY status, check the box below and enclose the PUBLICATION FEE and 1/2 the ISSUE FEE shown above.  
☐ Applicant claims SMALL ENTITY status.  
See 37 CFR 1.27.

II. PART B - FEE(S) TRANSMITTAL should be completed and returned to the United States Patent and Trademark Office (USPTO) with your ISSUE FEE and PUBLICATION FEE (if required). Even if the fee(s) have already been paid, Part B - Fee(s) Transmittal should be completed and returned. If you are charging the fee(s) to your deposit account, section "4b" of Part B - Fee(s) Transmittal should be completed and an extra copy of the form should be submitted.

III. All communications regarding this application must give the application number. Please direct all communications prior to issuance to Box ISSUE FEE unless advised to the contrary.

IMPORTANT REMINDER: Utility patents issuing on applications filed on or after Dec. 12, 1980 may require payment of maintenance fees. It is patentee's responsibility to ensure timely payment of maintenance fees when due.

77

PART B - FEE(S) TRANSMITTAL

Complete and send this form, together with applicable fee(s), to: **Mail** **Box ISSUE FEE**  
**Commissioner for Patents**  
**Washington, D.C. 20231**  
**Fax** **(703)746-4000**

INSTRUCTIONS: This form should be used for transmitting the ISSUE FEE and PUBLICATION FEE (if required). Blocks 1 through 4 should be completed where appropriate. All further correspondence including the Patent, advance orders and notification of maintenance fees will be mailed to the current correspondence address as indicated unless corrected below or directed otherwise in Block 1, by (a) specifying a new correspondence address; and/or (b) indicating a separate "FEE ADDRESS" for maintenance fee notifications.

CURRENT CORRESPONDENCE ADDRESS (Note: Legibly mark-up with any corrections or use Block 1)  
7590 03/26/2003

JORDAN M MESCHKOW  
MESCHKOW & GRESHAM PLC  
5727 NORTH SEVENTH STREET  
SUITE 409  
PHOENIX, AZ 85014

Note: A certificate of mailing can only be used for domestic mailings of the Fee(s) Transmittal. This certificate cannot be used for any other accompanying papers. Each additional paper, such as an assignment or formal drawing, must have its own certificate of mailing or transmission.

**Certificate of Mailing or Transmission**  
I hereby certify that this Fee(s) Transmittal is being deposited with the United States Postal Service with sufficient postage for first class mail in an envelope addressed to the Box Issue Fee address above, or being facsimile transmitted to the USPTO, on the date indicated below.

(Depositor's name)
(Signature)
(Date)

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
09/429,357	10/28/1999	CHARLES P. MCCOLLUM	7791-A-1	5829

TITLE OF INVENTION: METHOD AND SYSTEM FOR ADDING FUNCTION TO A WEB PAGE

APPLN. TYPE	SMALL ENTITY	ISSUE FEE	PUBLICATION FEE	TOTAL FEE(S) DUE	DATE DUE
nonprovisional	YES	\$650	\$0	\$650	06/26/2003

EXAMINER	ART UNIT	CLASS-SUBCLASS
CLINTON, GREGORY L	2154	709-218000

1. Change of correspondence address or indication of "Fee Address" (37 CFR 1.363). <input type="checkbox"/> Change of correspondence address (or Change of Correspondence Address form PTO/SB/122) attached. <input type="checkbox"/> "Fee Address" indication (or "Fee Address" Indication form PTO/SB/47; Rev 03-02 or more recent) attached. <b>Use of a Customer Number is required.</b>	2. For printing on the patent front page, list (1) the names of up to 3 registered patent attorneys or agents OR, alternatively, (2) the name of a single firm (having as a member a registered attorney or agent) and the names of up to 2 registered patent attorneys or agents. If no name is listed, no name will be printed. 1 _____ 2 _____ 3 _____
--	--

3. ASSIGNEE NAME AND RESIDENCE DATA TO BE PRINTED ON THE PATENT (print or type)

PLEASE NOTE: Unless an assignee is identified below, no assignee data will appear on the patent. Inclusion of assignee data is only appropriate when an assignment has been previously submitted to the USPTO or is being submitted under separate cover. Completion of this form is NOT a substitute for filing an assignment.

(A) NAME OF ASSIGNEE (B) RESIDENCE: (CITY and STATE OR COUNTRY)

Please check the appropriate assignee category or categories (will not be printed on the patent) ☐ individual ☐ corporation or other private group entity ☐ government

4a. The following fee(s) are enclosed: <input type="checkbox"/> Issue Fee <input type="checkbox"/> Publication Fee <input type="checkbox"/> Advance Order - # of Copies _____	4b. Payment of Fee(s): <input type="checkbox"/> A check in the amount of the fee(s) is enclosed. <input type="checkbox"/> Payment by credit card. Form PTO-2038 is attached. <input type="checkbox"/> The Commissioner is hereby authorized by charge the required fee(s), or credit any overpayment, to Deposit Account Number _____ (enclose an extra copy of this form).
--	--

Commissioner for Patents is requested to apply the Issue Fee and Publication Fee (if any) or to re-apply any previously paid issue fee to the application identified above.

(Authorized Signature)	(Date)
NOTE: The Issue Fee and Publication Fee (if required) will not be accepted from anyone other than the applicant; a registered attorney or agent; or the assignee or other party in interest as shown by the records of the United States Patent and Trademark Office.	
This collection of information is required by 37 CFR 1.311. The information is required to obtain or retain a benefit by the public which is to file (and by the USPTO to process) an application. Confidentiality is governed by 35 U.S.C. 122 and 37 CFR 1.14. This collection is estimated to take 12 minutes to complete, including gathering, preparing, and submitting the completed application form to the USPTO. Time will vary depending upon the individual case. Any comments on the amount of time you require to complete this form and/or suggestions for reducing this burden, should be sent to the Chief Information Officer, U.S. Patent and Trademark Office, U.S. Department of Commerce, Washington, D.C. 20231. DO NOT SEND FEES OR COMPLETED FORMS TO THIS ADDRESS. SEND TO: Commissioner for Patents, Washington, DC 20231.	
Under the Paperwork Reduction Act of 1995, no persons are required to respond to a collection of information unless it displays a valid OMB control number.	



UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE  
United States Patent and Trademark Office  
Address: COMMISSIONER OF PATENTS AND TRADEMARKS  
Washington, D.C. 20231  
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
09/429,357	10/28/1999	CHARLES P. MCCOLLUM	7791-A-1	5829
7590 03/26/2003				
JORDAN M MESCHKOW MESCHKOW & GRESHAM PLC 5727 NORTH SEVENTH STREET SUITE 409 PHOENIX, AZ 85014			EXAMINER  CLINTON, GREGORY L	
			ART UNIT	PAPER NUMBER
			2154	
DATE MAILED: 03/26/2003				

Determination of Patent Term Extension under 35 U.S.C. 154 (b)  
(application filed after June 7, 1995 but prior to May 29, 2000)

The patent term extension is 0 days. Any patent to issue from the above identified application will include an indication of the 0 day extension on the front page.

If a continued prosecution application (CPA) was filed in the above-identified application, the filing date that determines patent term extension is the filing date of the most recent CPA.

Applicant will be able to obtain more detailed information by accessing the Patent Application Information Retrieval (PAIR) system. (<http://pair.uspto.gov>)

Any questions regarding the patent term extension or adjustment determination should be directed to the Office of Patent Legal Administration at (703)305-1383.





## UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE  
 United States Patent and Trademark Office  
 Address: COMMISSIONER OF PATENTS AND TRADEMARKS  
 Washington, D.C. 20231  
[www.uspto.gov](http://www.uspto.gov)

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
09/429,357	10/28/1999	CHARLES P. MCCOLLUM	7791-A-1	5829
7590	03/26/2003		EXAMINER	
JORDAN M MESCHKOW MESCHKOW & GRESHAM PLC 5727 NORTH SEVENTH STREET SUITE 409 PHOENIX, AZ 85014			CLINTON, GREGORY L	
			ART UNIT	PAPER NUMBER
			2154	
DATE MAILED: 03/26/2003				

### Notice of Fee Increase on January 1, 2003

If a reply to a "Notice of Allowance and Fee(s) Due" is filed in the Office on or after January 1, 2003, then the amount due will be higher than that set forth in the "Notice of Allowance and Fee(s) Due" since there will be an increase in fees effective on January 1, 2003. See Revision of Patent and Trademark Fees for Fiscal Year 2003; Final Rule, 67 Fed. Reg. 70847, 70849 (November 27, 2002).

The current fee schedule is accessible from: <http://www.uspto.gov/main/howtofees.htm>.

If the issue fee paid is the amount shown on the "Notice of Allowance and Fee(s) Due," but not the correct amount in view of the fee increase, a "Notice to Pay Balance of Issue Fee" will be mailed to applicant. In order to avoid processing delays associated with mailing of a "Notice to Pay Balance of Issue Fee," if the response to the Notice of Allowance and Fee(s) due form is to be filed on or after January 1, 2003 (or mailed with a certificate of mailing on or after January 1, 2003), the issue fee paid should be the fee that is required at the time the fee is paid. If the issue fee was previously paid, and the response to the "Notice of Allowance and Fee(s) Due" includes a request to apply a previously-paid issue fee to the issue fee now due, then the difference between the issue fee amount at the time the response is filed and the previously paid issue fee should be paid. See Manual of Patent Examining Procedure, Section 1308.01 (Eighth Edition, August 2001).

Questions relating to issue and publication fee payments should be directed to the Customer Service Center of the Office of Patent Publication at (703) 305-8283.



IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

Applicant: Charles P. McCollum, et al.

Serial No.: 09/429,357

Filed: 28 October 1999

For: *METHOD AND SYSTEM FOR ADDING FUNCTION TO A WEB PAGE*

**CERTIFICATE OF MAILING**

Mail Stop Issue Fee  
Commissioner for Patents  
P.O. Box 1450  
Alexandria, Virginia 22313-1450

Dear Sir:

I hereby certify that this correspondence, consisting of this Certificate of Mailing; Issue Fee Transmittal; Check in the amount of \$680.00 (Issue Fee and copies); Eleven (11) sheets of drawings and 3 copies of same; and a Postcard, are being deposited with the United States Postal Service with sufficient postage as first class mail in an envelope addressed to:

Mail Stop Issue Fee  
Commissioner for Patents  
P.O. Box 1450  
Alexandria, Virginia 22313-1450

on

26 May 2003  
Date

26 May 2003

MESCHKOW & GRESHAM, P.L.C.  
5727 North Seventh Street  
Suite 409  
Phoenix, Arizona 85014  
(602) 274-6996

[Signature]  
Signature

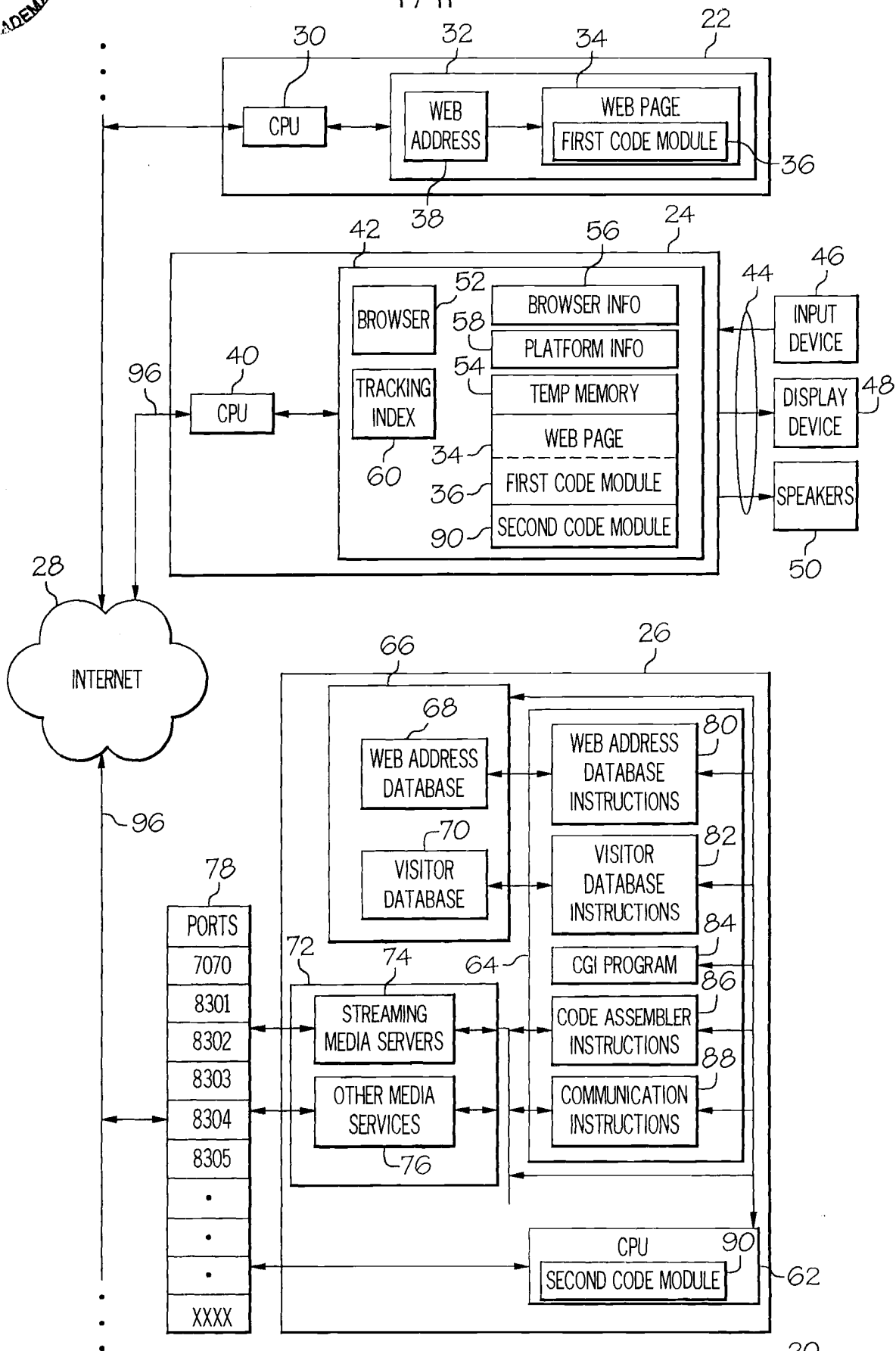
Respectfully submitted,  
[Signature]  
Jordan M. Meschkow  
Attorney for Applicant  
Registration No. 31,043



09.4/29.357

6594691

1 / 11



20



2 / 11

LINE NO.	93 CODE
92 1	<script src= 'http://bslserver.domainname.com/ cgi-bin/bslservercall.cgi' > 94
98 2	</script>
100 3	<script><!-- 102
104 4	BSLStart (); 106
108 5	//--></script> 102

36

FIG. 2



3 / 11

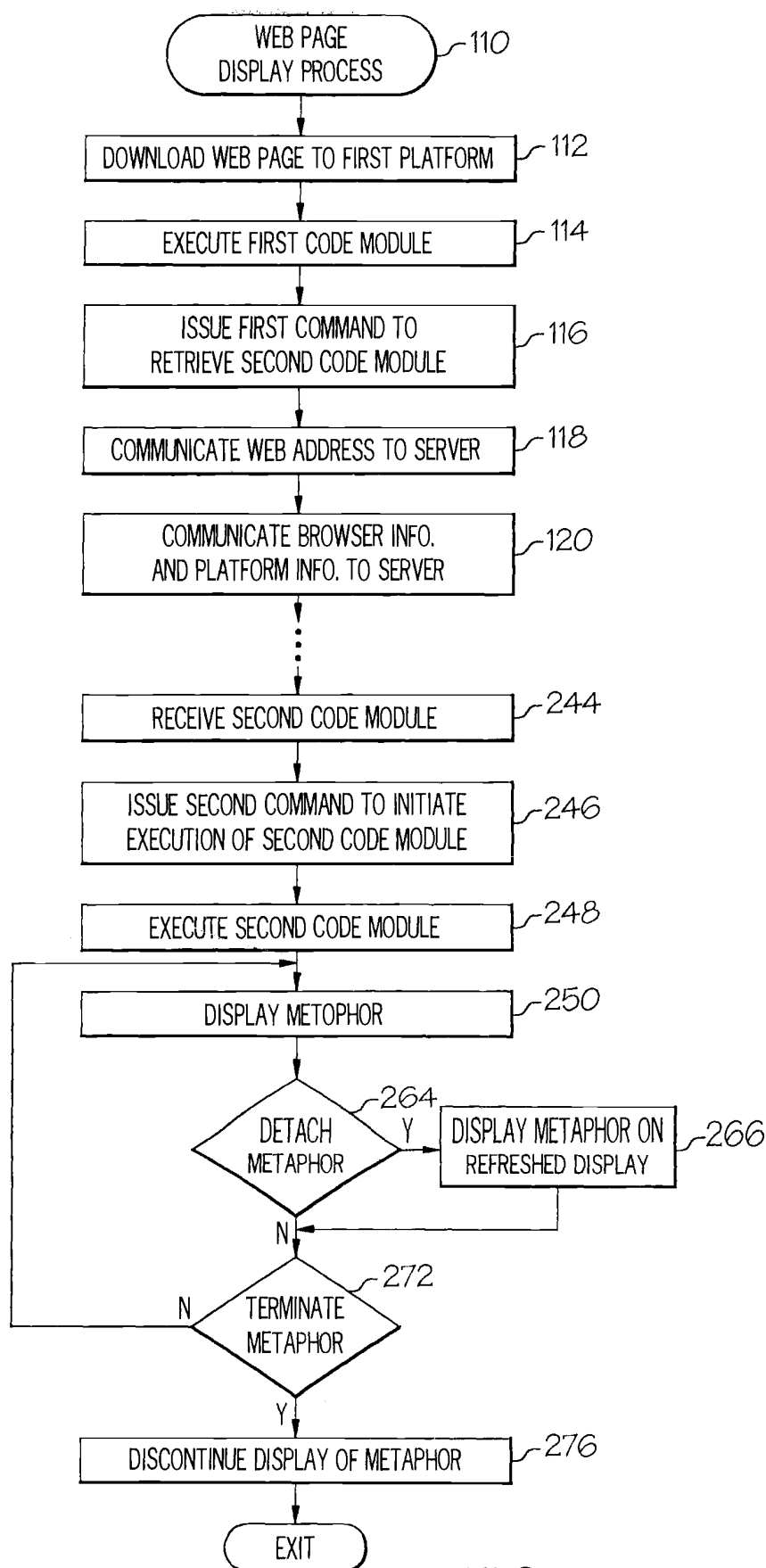


FIG. 3



4 / 11

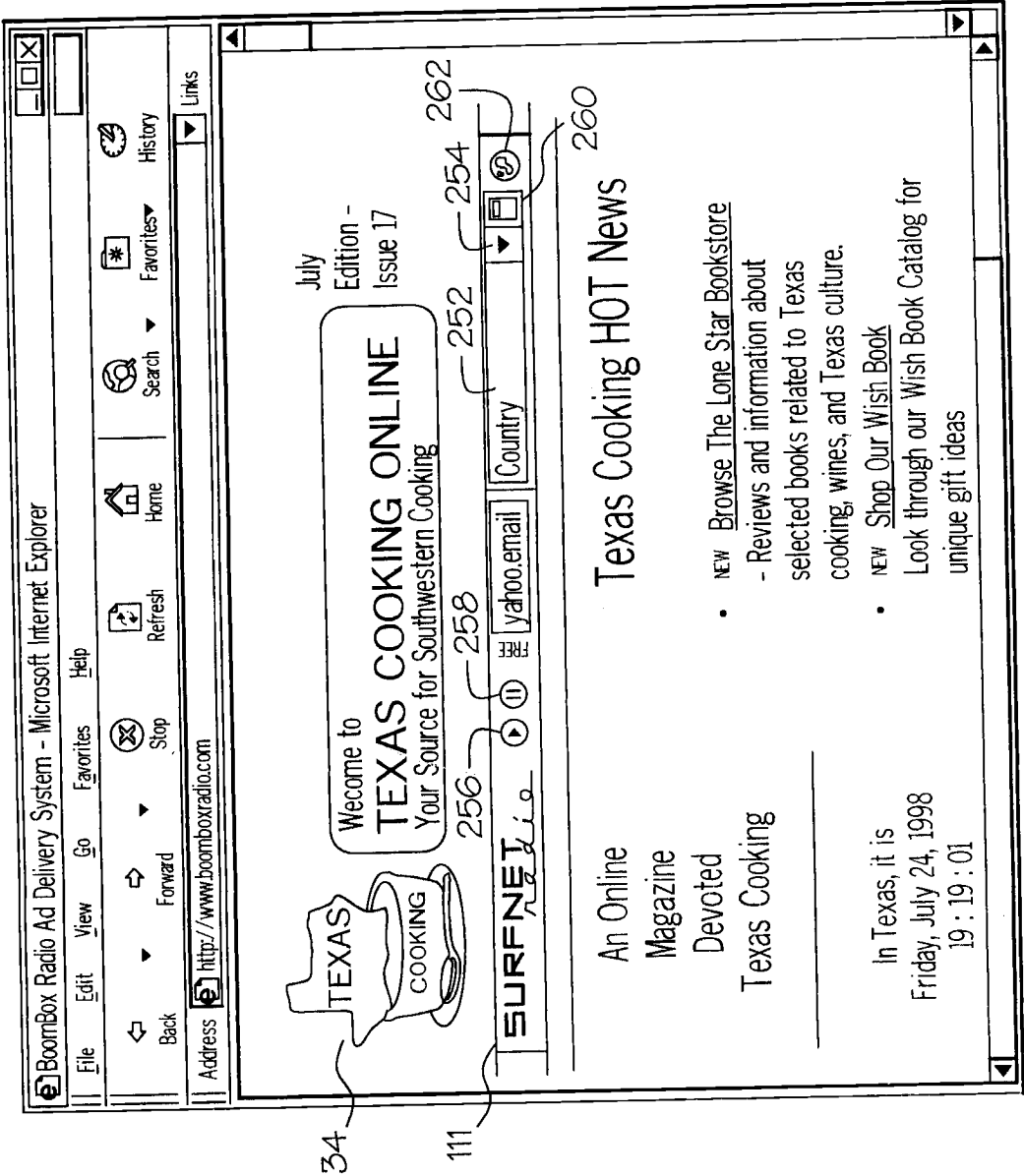


FIG. 4

48



5 / 11

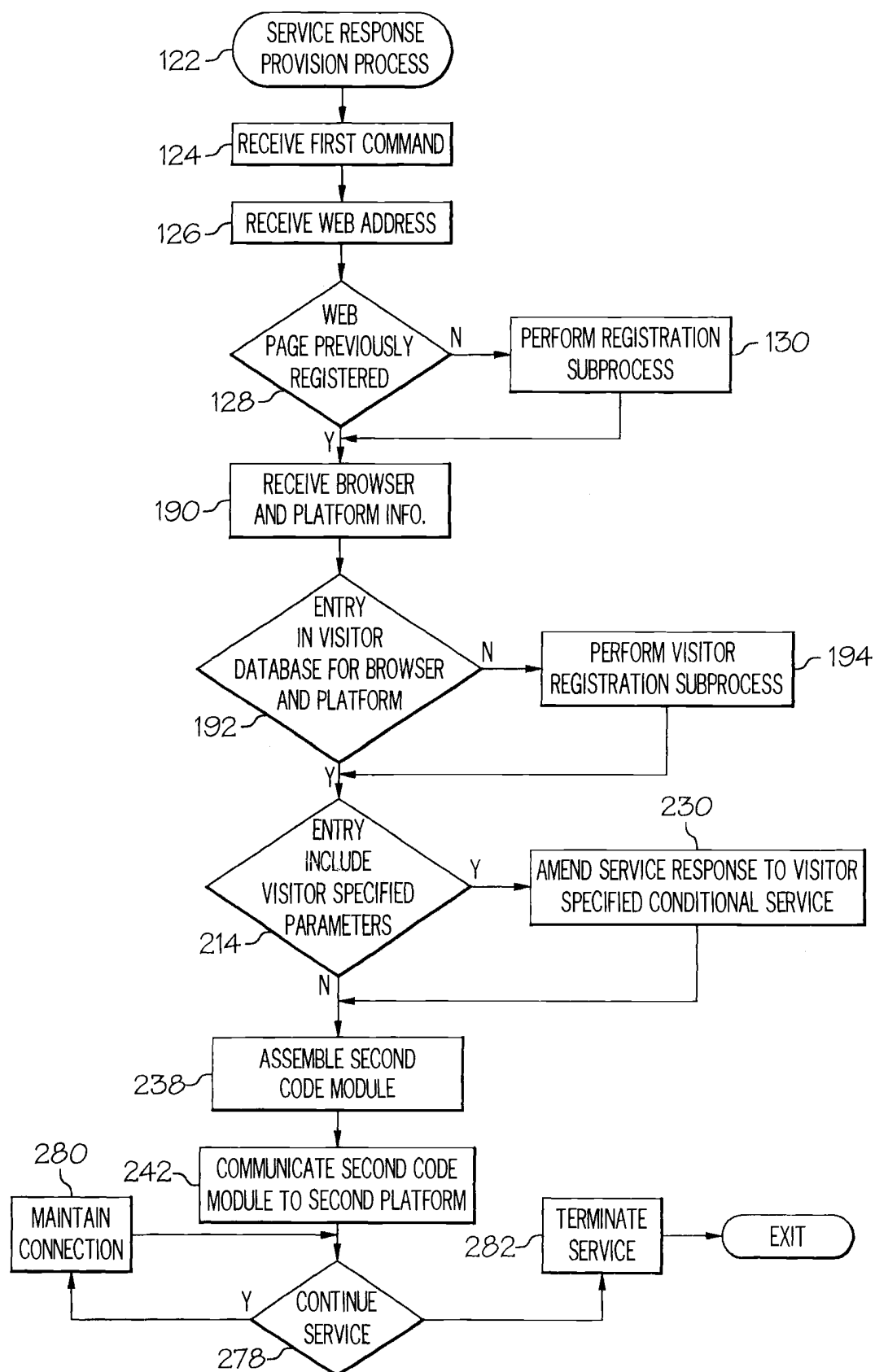


FIG. 5



6 / 11

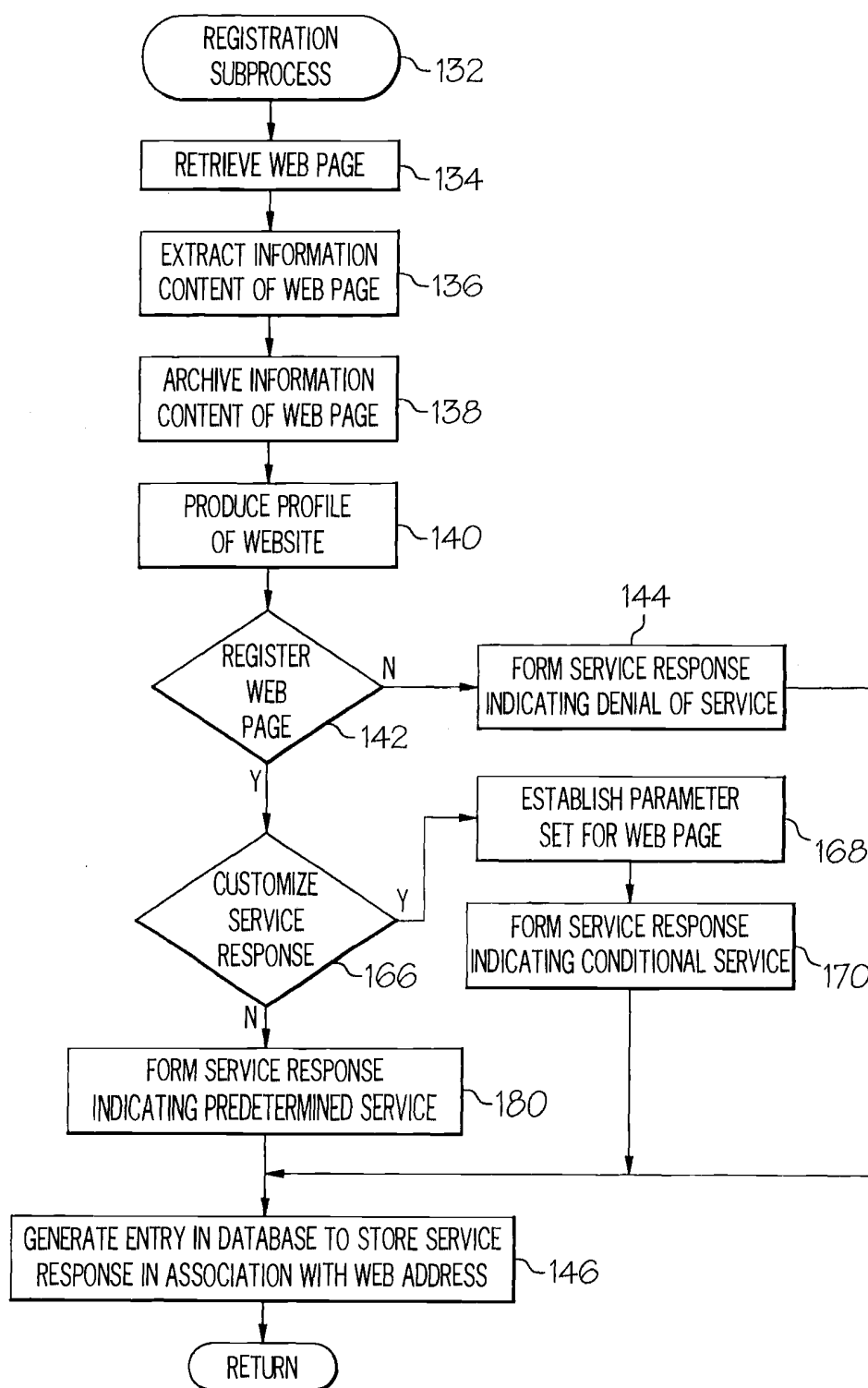


FIG. 6





7 / 11

	150 WEB ADDRESS FIELD	152 PROFILE FIELD	154 SERVICE RESPONSE 162 FIELD	PARAMETER SET FIELD	156
158	URL 1	RECREATION/ GOLF	DENIAL OF 176 SERVICE	DENIAL CONTENT	164
172	URL 2	TEXAS COOKING	CONDITIONAL 186 SERVICE	CONDITIONAL CONTENT (INCLUDING URL 5)	178
182	URL 3	WEDDING	PREDETERMINED 186 SERVICE	PREDETERMINED CONTENT	188
232	URL 4	FOOTBALL 234	PREDETERMINED SERVICE (FLAG- CONDITIONAL SERVICE FOR TRACKING INDEX 60)	PREDETERMINED CONTENT	188
	⋮		⋮		
	URL n		.		

FIG. 7

68

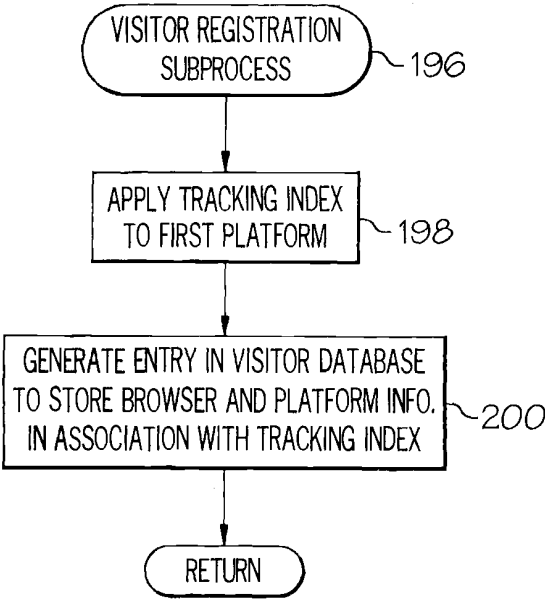


FIG. 8



	202	204	206	
	TRACKING INDEX	BROWSER ID	PLATFORM ID	VISITOR PREFERENCES
210	60	60	60	208
	SECOND PLATFORM	BROWSER INFO	PLATFORM INFO	VISITOR SPECIFIED PATAMETER SET
		56	58	212

FIG. 9

70



9 / 11

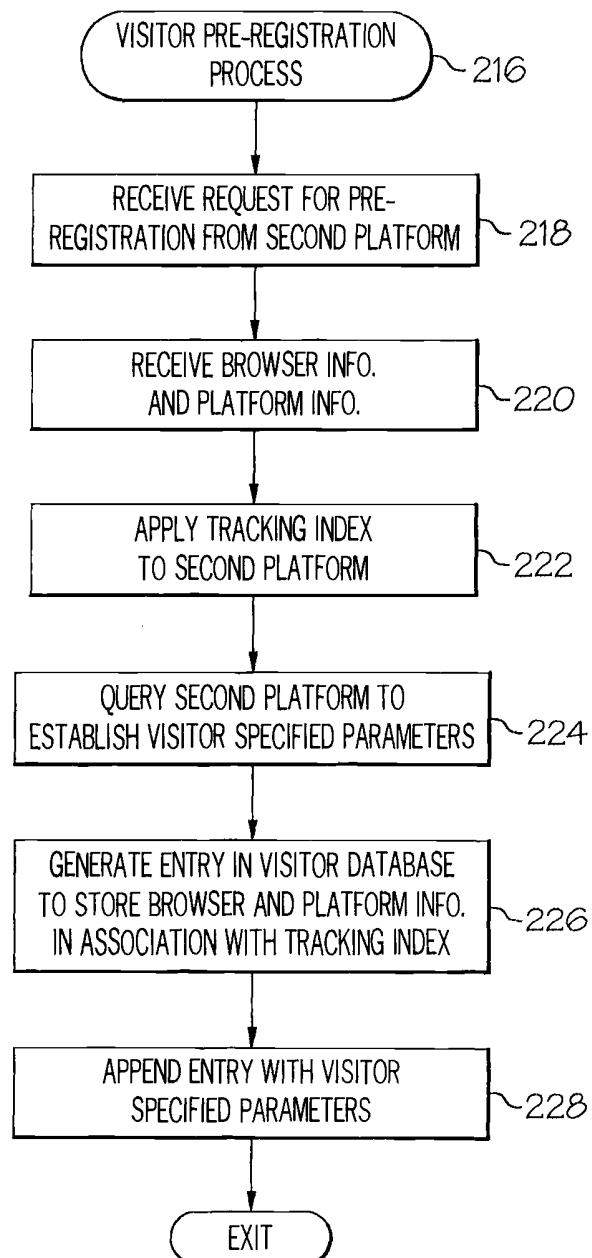
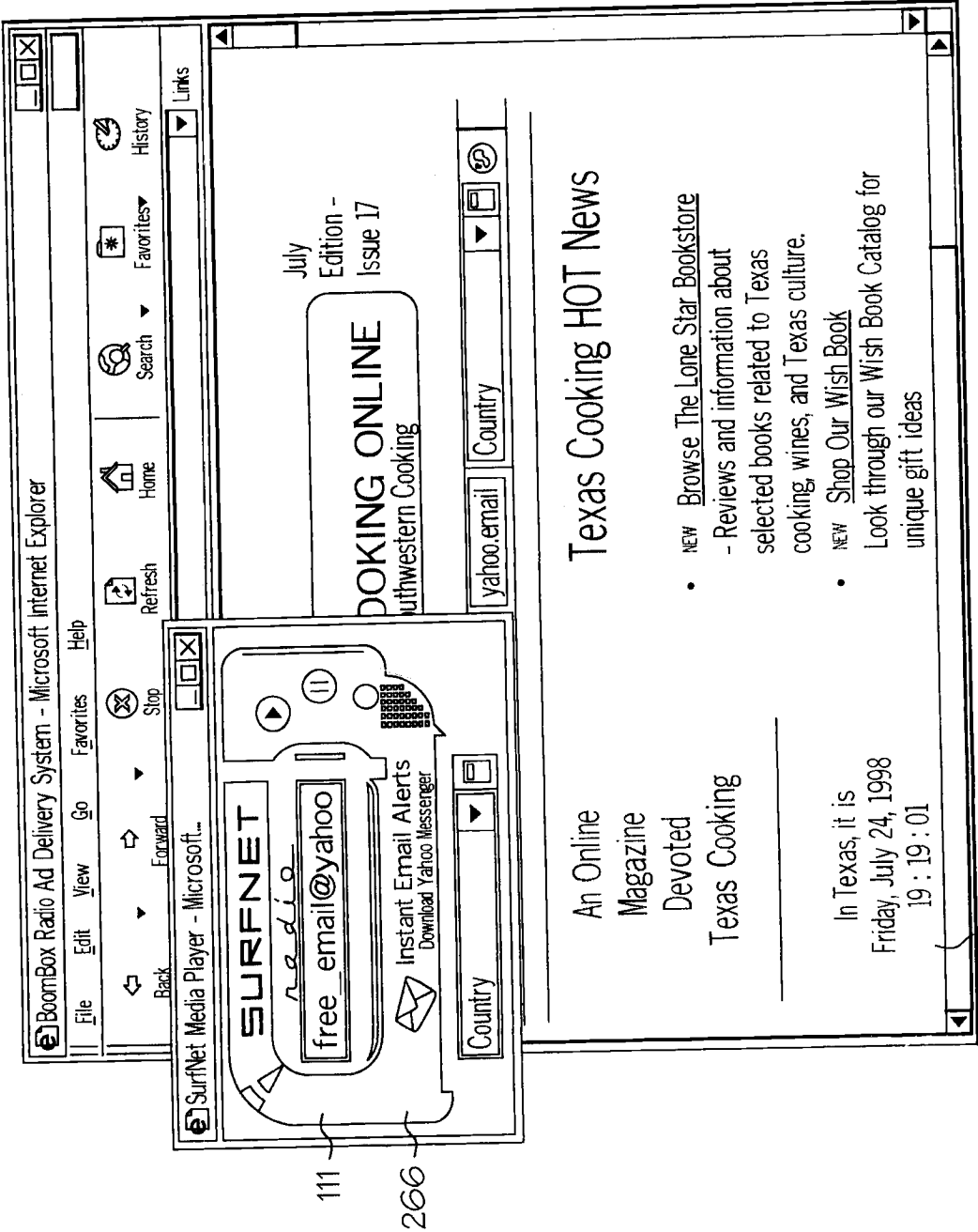


FIG. 10



10 / 11



48

FIG. 11

34

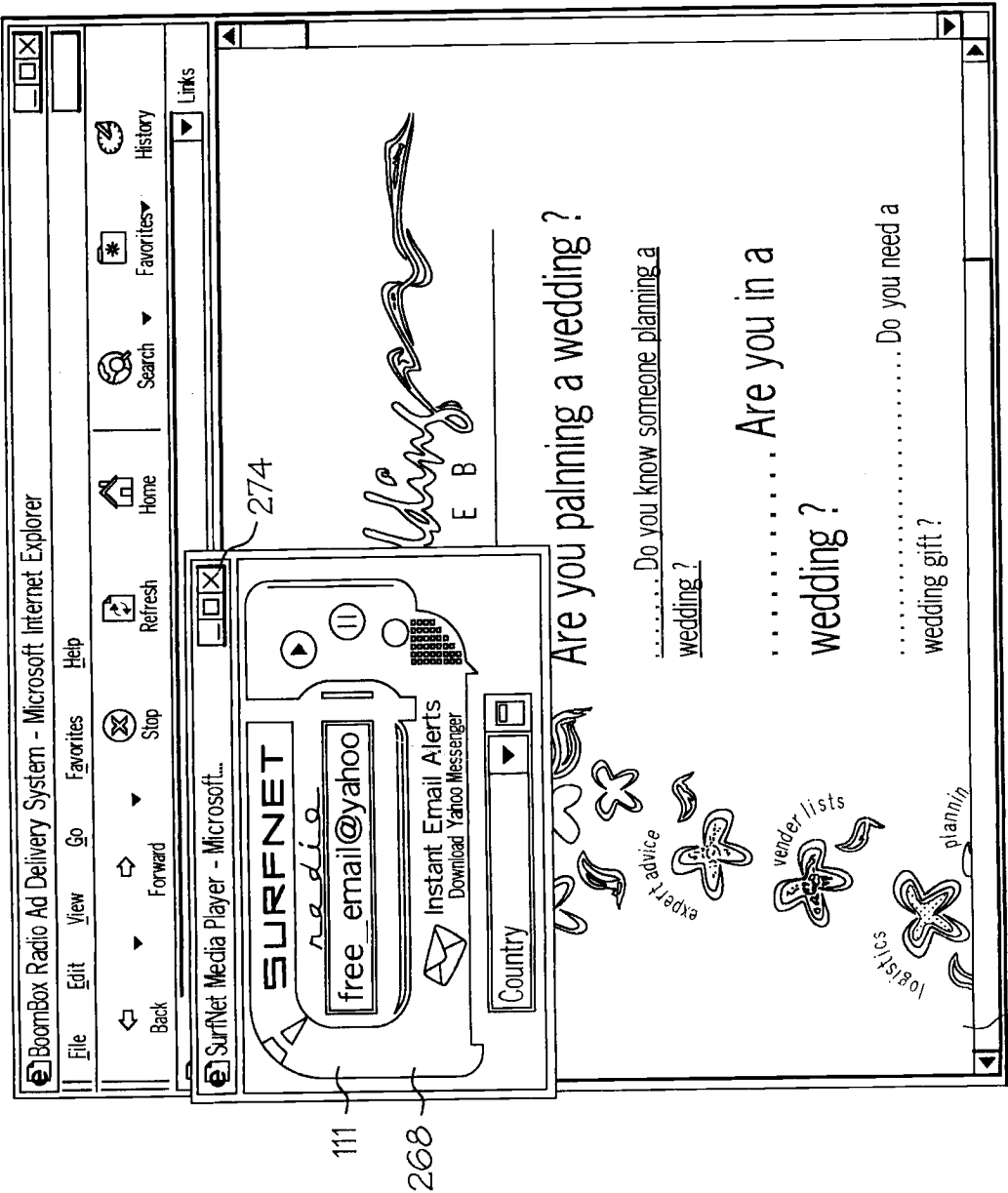


FIG. 12

## PART B - FEE(S) TRANSMITTAL

Complete and send this form, together with applicable fee(s), to: **Mail** Box ISSUE FEE  
 Commissioner for Patents  
 Washington, D.C. 20231  
**Fax** (703)746-4000

**INSTRUCTIONS:** This form should be used for transmitting the ISSUE FEE and PUBLICATION FEE (if required). Blocks 1 through 4 should be completed where appropriate. All further correspondence including the Patent, advance orders and notification of maintenance fees will be mailed to the current correspondence address as indicated unless corrected below or directed otherwise in Block 1, by (a) specifying a new correspondence address; and/or (b) indicating a separate "FEE ADDRESS" for maintenance fee notifications.

CURRENT CORRESPONDENCE ADDRESS (Note: Legibly mark-up with any corrections to use Block 1)  
 7590 03/26/2003

JORDAN M MESCHKOW  
 MESCHKOW & GRESHAM PLC  
 5727 NORTH SEVENTH STREET  
 SUITE 409  
 PHOENIX, AZ 85014



Note: A certificate of mailing can only be used for domestic mailings of the Fee(s) Transmittal. This certificate cannot be used for any other accompanying papers. Each additional paper, such as an assignment or formal drawing, must have its own certificate of mailing or transmission.

**Certificate of Mailing or Transmission**  
 I hereby certify that this Fee(s) Transmittal is being deposited with the United States Postal Service with sufficient postage for first class mail in an envelope addressed to the Box Issue Fee address above, or being facsimile transmitted to the USPTO, on the date indicated below.

Jordan M. Meschkow (Depositor's name)  
 [Signature] (Signature)  
 26 May 2003 (Date)

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
09/429,357	10/28/1999	CHARLES F. MCCOLLUM	7791-A-1	3829

TITLE OF INVENTION: METHOD AND SYSTEM FOR ADDING FUNCTION TO A WEB PAGE

APPL. TYPE	SMALL ENTITY	ISSUE FEE	PUBLICATION FEE	TOTAL FEE(S) DUE	DATE DUE
nonprovisional	YES	\$650	\$0	\$650	06/26/2003

EXAMINER	ART UNIT	CLASS-SUBCLASS
CLINTON, GREGORY L	2154	709-218000

1. Change of correspondence address or indication of "Fee Address" (37 CFR 1.303).  
☐ Change of correspondence address (or Change of Correspondence Address form PTO/SB/122) attached.  
☐ "Fee Address" indication (or "Fee Address" Indication form PTO/SB/47; Rev 03-02 or more recent) attached. Use of a Customer Number is required.

2. For printing on the patent front page, list (1) the names of up to 3 registered patent attorneys or agents OR, alternatively, (2) the name of a single firm (having as a member a registered attorney or agent) and the names of up to 2 registered patent attorneys or agents. If no name is listed, no name will be printed.

1 Jordan M. Meschkow  
 2 Lowell W. Gresham  
 3 Charlene R. Jacobsen

### 3. ASSIGNEE NAME AND RESIDENCE DATA TO BE PRINTED ON THE PATENT (print or type)

PLEASE NOTE: Unless an assignee is identified below, no assignee data will appear on the patent. Inclusion of assignee data is only appropriate when an assignment has been previously submitted to the USPTO or is being submitted under separate cover. Completion of this form is NOT a substitute for filing an assignment.

(A) NAME OF ASSIGNEE (B) RESIDENCE: (CITY AND STATE OR COUNTRY)

Surfnet Media Group, Inc.

Tempe, Arizona

Please check the appropriate assignee category or categories (will not be printed on the patent) ☐ individual ☒ corporation or other private group entity ☐ government

#### 4a. The following fee(s) are enclosed:

☒ Issue Fee  
☐ Publication Fee

☒ Advance Order - # of Copies 10

#### 4b. Payment of Fee(s):

☒ A check in the amount of the fee(s) is enclosed.

☐ Payment by credit card. Form PTO-2038 is attached.

☐ The Commissioner is hereby authorized by charge the required fee(s), or credit any overpayment, to Deposit Account Number (enclose an extra copy of this form).

Commissioner for Patents is requested to apply the Issue Fee and Publication Fee (if any) or to re-apply any previously paid issue fee to the application identified above.

(Authorized Signature)

(Date)

NOTE: The Issue Fee and Publication Fee (if required) will not be accepted from anyone other than the applicant, a registered attorney or agent, or the assignee or other party in interest as shown by the records of the United States Patent and Trademark Office.

This collection of information is required by 37 CFR 1.311. The information is required to obtain or retain a benefit by the public which is to file (and by the USPTO to process) an application. Confidentiality is governed by 35 U.S.C. 122 and 37 CFR 1.14. This collection is estimated to take 12 minutes to complete, including gathering, preparing, and submitting the completed application form to the USPTO. Time will vary depending upon the individual case. Any comments on the amount of time you require to complete this form and/or suggestions for reducing this burden, should be sent to the Chief Information Officer, U.S. Patent and Trademark Office, U.S. Department of Commerce, Washington, D.C. 20231. DO NOT SEND FEES OR COMPLETED FORMS TO THIS ADDRESS. SEND TO: Commissioner for Patents, Washington, DC 20231.

Under the Paperwork Reduction Act of 1995, no persons are required to respond to a collection of information unless it displays a valid OMB control number.

06/03/2003 JMD:LMR 00000000 09429357

01 FC:2501  
 02 FC:0001

650.00 BP  
 20.00 BP

TRANSMIT THIS FORM WITH FEE(S)

PTOL-85 (REV. 04-02) Approved for use through 01/31/2004. OMB 0651-0033

U.S. Patent and Trademark Office; U.S. DEPARTMENT OF COMMERCE



IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

Applicant: Charles P. McCollum, et al. )  
Serial No.: 09/429,357 )  
Filed: 28 October 1999 )  
For: METHOD AND SYSTEM FOR ADDING FUNCTION TO A WEB PAGE )

CERTIFICATE OF MAILING

Commissioner for Patents  
P.O. Box 1450  
Alexandria, Virginia 22313-1450

Certificate

AUG 08 2003

of Correction

Dear Sir:

I hereby certify that this correspondence, consisting of this Certificate of Mailing; Certificate of Correction of Office Mistake; Certificate of Correction form PTO/SB/44; and a Postcard, are being deposited with the United States Postal Service with sufficient postage as first class mail in an envelope addressed to:

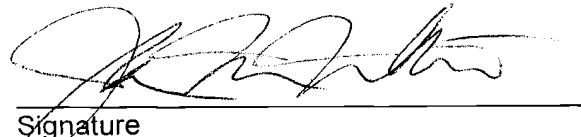
Commissioner for Patents  
P.O. Box 1450  
Alexandria, Virginia 22313-1450

on

1 August 2003  
Date

1 August 2003

MESCHKOW & GRESHAM, P.L.C.  
5727 North Seventh Street  
Suite 409  
Phoenix, Arizona 85014  
(602) 274-6996

  
Signature

Respectfully submitted,

  
Jordan M. Meschkow  
Attorney for Applicant  
Registration No. 31,043



IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

Applicant: McCollum, et al.	)	
	)	
Serial No.: 09/429,357	)	Ex: Zarni Maung
	)	
Filed: 28 October 1999	)	Art Unit: 2154
	)	
Patent No.: 6,594,691	)	Date: 1 August 2003
	)	
For: METHOD AND SYSTEM FOR	)	
ADDING FUNCTION TO A WEB	)	
PAGE	)	

**CERTIFICATE OF CORRECTION OF OFFICE MISTAKE**

Commissioner for Patents  
P.O. Box 1450  
Alexandria, Virginia 22313-1450

Dear Sir:

This is a request for a Certificate of Correction in accordance with the provisions of 37 C.F.R. 1.322.

A review of the above-identified patent discloses a certain error which affects the scope or meaning thereof. The error is of a minor character, the correction of which does not constitute new matter nor would require reexamination.

The error and proposed correction is:

In Column 1, Line 57: correct the spelling of the word --propiety-- insert the word --proprietary-- after the words "media if it is" at the end of the sentence.

In Column 7, Line 32: insert the words --(such as .com, .edu, .jp, .uk, etc.),— and delete the words --(such as com, .edu., .jp, .uk, etc.),— after the words "domain name extensions" and before the words "and so forth."

In Column 11, Line 58: insert the words --specified parameter— and delete the words --specified; parameter— after the words "include visitor" and before the words "set 212, process"

In Column 13, Line 45: insert the words --subsequent web page.— and delete the words --subsequent. Web page.— after the words "and downloads a" at the end of the sentence.

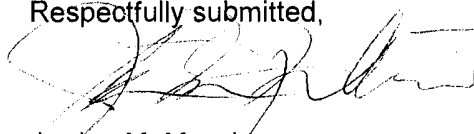


A completed Certificate of Correction, Form PTO/SB/44, setting forth the above is enclosed.

The foregoing correction(s) bring(s) the wordage of the patent into accord with the wordage provided by Patentee, either in the original application or amendments thereto. Accordingly, it is submitted that the mistake is by way of the fault of the Office and that the Certificate of Correction should be issued without charge to the Patentee.

Patentee respectfully solicits the granting of this request.

Respectfully submitted,

A handwritten signature in black ink, appearing to read 'Jordan M. Meschkow', is written over a horizontal line.

Jordan M. Meschkow  
Attorney for Applicant  
Registration No. 31,043

PTO/SB/44 (07-03)

Approved for use through 01/31/2004. OMB 0651-0033

U.S. Patent and Trademark Office; U.S. DEPARTMENT OF COMMERCE

Under the Paperwork Reduction Act of 1995, no persons are required to respond to a collection of information unless it displays a valid OMB control number.

(Also Form PTO-1050)

## UNITED STATES PATENT AND TRADEMARK OFFICE CERTIFICATE OF CORRECTION

PATENT NO. : 6,594,691

DATED : 15 July 2003

INVENTOR(S) : McCollum, et al.

It is certified that error appears in the above-identified patent and that said Letters Patent is hereby corrected as shown below:

In Column 1, Line 57: correct the spelling of the word --propiety-- insert the word --proprietary-- after the words "media if it is" at the end of the sentence.

In Column 7, Line 32: Insert the words --(such as .com, .edu, .jp, .uk, etc.),-- and delete the words -- (such as com, .edu., .jp, .uk, etc.),-- after the words "domain name extensions" and before the words "and so forth."

In Column 11, Line 58: Insert the words --specified parameter-- and delete the words --specified; parameter-- after the words "include visitor" and before the words "set 212, process"

In Column 13, Line 45: Insert the words --subsequent web page.-- and delete the words --subsequent. Web page.-- after the words "and downloads a" at the end of the sentence.

## MAILING ADDRESS OF SENDER:

Jordan M. Meschkow

MESCHKOW &amp; GRESHAM, PLC

5727 N. 7th Street, Suite 409, Phoenix, Arizona 85014 ➡

PATENT NO. 6,594,691

No. of additional copies

This collection of information is required by 37 CFR 1.322, 1.323, and 1.324. The information is required to obtain or retain a benefit by the public which is to file (and by the USPTO to process) an application. Confidentiality is governed by 35 U.S.C. 122 and 37 CFR 1.14. This collection is estimated to take 1.0 hour to complete, including gathering, preparing, and submitting the completed application form to the USPTO. Time will vary depending upon the individual case. Any comments on the amount of time you require to complete this form and/or suggestions for reducing this burden, should be sent to the Chief Information Officer, U.S. Patent and Trademark Office, U.S. Department of Commerce, P.O. Box 1450, Alexandria, VA 22313-1450. DO NOT SEND FEES OR COMPLETED FORMS TO THIS ADDRESS. SEND TO: Attention Certificate of Corrections Branch, Commissioner for Patents, P.O. Box 1450, Alexandria, VA 22313-1450.

If you need assistance in completing the form, call 1-800-PTO-9199 and select option 2.

AUG 12 2003

UNITED STATES PATENT AND TRADEMARK OFFICE  
**CERTIFICATE OF CORRECTION**

PATENT NO. : 6,594,691 B1  
DATED : July 15, 2003  
INVENTOR(S) : McCollum et al.

Page 1 of 1

It is certified that error appears in the above-identified patent and that said Letters Patent is hereby corrected as shown below:

Column 7.

Line 32, insert the words -- (such as .com, .edu, .jp, .uk, etc.), -- and delete the words -- (such as com, .edu., .jp, .uk, etc.), -- after the words "domain name extensions" and before the words "and so forth."

Column 11.

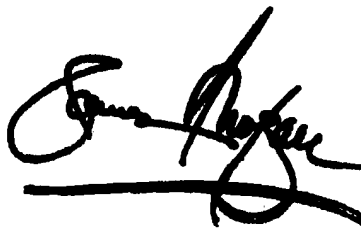
Line 58, insert the words -- specified parameter -- and delete the words -- specified; parameter -- after the words "include visitor" and before the words "set 212, process"

Column 13.

Line 45, insert the words -- subsequent web page. -- and delete the words -- subsequent. Web page. -- after the words "and downloads a" at the end of the sentence.

Signed and Sealed this

Fourteenth Day of October, 2003

A handwritten signature in black ink, appearing to read "James E. Rogan", with a horizontal line drawn underneath it.

JAMES E. ROGAN  
*Director of the United States Patent and Trademark Office*

PTO/SB/44 (07-03)

Approved for use through 01/31/2004. OMB 0651-0033

U.S. Patent and Trademark Office; U.S. DEPARTMENT OF COMMERCE

Under the Paperwork Reduction Act of 1995, no persons are required to respond to a collection of information unless it displays a valid OMB control number.  
(Also Form PTO-1050)

# UNITED STATES PATENT AND TRADEMARK OFFICE CERTIFICATE OF CORRECTION

PATENT NO. : 6,594,691 **B1**  
 DATED : 15 July 2003  
 INVENTOR(S) : McCollum, et al.

It is certified that error appears in the above-identified patent and that said Letters Patent is hereby corrected as shown below:

**No** In Column 1, Line 57: correct the spelling of the word --propiety-- insert the word --proprietary-- after the words "media if it is" at the end of the sentence.

In Column 7, Line 32: Insert the words --(such as .com, .edu, .jp, .uk, etc.),-- and delete the words -- (such as com, .edu., .jp, .uk, etc.),-- after the words "domain name extensions" and before the words "and so forth."

In Column 11, Line 58: Insert the words --specified parameter-- and delete the words --specified; parameter-- after the words "include visitor" and before the words "set 212, process"

In Column 13, Line 45: Insert the words --subsequent web page.-- and delete the words --subsequent. Web page.-- after the words "and downloads a" at the end of the sentence.

## MAILING ADDRESS OF SENDER:

Jordan M. Meschkow  
 MESCHKOW & GRESHAM, PLC

5727 N. 7th Street, Suite 409, Phoenix, Arizona 85014 →

PATENT NO. 6,594,691

No. of additional copies

This collection of information is required by 37 CFR 1.322, 1.323, and 1.324. The information is required to obtain or retain a benefit by the public which is to file (and by the USPTO to process) an application. Confidentiality is governed by 35 U.S.C. 122 and 37 CFR 1.14. This collection is estimated to take 1.0 hour to complete, including gathering, preparing, and submitting the completed application form to the USPTO. Time will vary depending upon the individual case. Any comments on the amount of time you require to complete this form and/or suggestions for reducing this burden, should be sent to the Chief Information Officer, U.S. Patent and Trademark Office, U.S. Department of Commerce, P.O. Box 1450, Alexandria, VA 22313-1450. DO NOT SEND FEES OR COMPLETED FORMS TO THIS ADDRESS. SEND TO: Attention Certificate of Corrections Branch, Commissioner for Patents, P.O. Box 1450, Alexandria, VA 22313-1450.

If you need assistance in completing the form, call 1-800-PTO-9199 and select option 2.

AUG 12 2003



UNITED STATES PATENT AND TRADEMARK OFFICE

COMMISSIONER FOR PATENTS  
UNITED STATES PATENT AND TRADEMARK OFFICE  
P.O. Box 1450  
ALEXANDRIA, VA 22313-1450  
www.uspto.gov

Date : September 23, 2003  
Patent No : 6,594,691  
Inventor : McCollum, et al.  
Issued : July 15, 2003  
For : METHOD AND SYSTEM FOR ADDING FUNCTION TO A WEB PAGE

# 9  
spen

Re: Request for Certificate of Correction

Consideration has been given your request for the issuance of a certificate of correction for the above-identified patent under the provisions of Rule 1.322.

Respecting the alleged error(s) in Col. 1, line 57, after inspection of the application, it is revealed that the line is printed in accordance with the record. Therefore, being only partially the fault of the Patent and Trademark Office, partial correction(s) are in order here under United States Codes (U.S.C.) 254 or 255 and the Code of Federal Regulation (C.F.R.) 1.322 or 1.323).

In view of the foregoing, your request in this matter is hereby partially denied.

A certificate will issue for the remaining errors in your request.

Further consideration will be given in *this matter*, upon receipt of a reconsideration, with its corresponding fee of \$100.00, and supporting documentation.

*Note: for future reference, if you obtain data that clearly shows the attorney/applicant is not in error, simply include proving documentation along with the certificate of correction or reconsideration; this may decrease the processing time of your request.*

A handwritten signature in black ink, appearing to read "Stacy Powell, Sr.".

Stacy Powell, Sr.  
For Cecelia B. Newman, Supervisor  
Decisions & Certificates of  
Correction Branch  
(703) 605-5248 or (703) 305-8309

Jordan M. Meschkow  
MESCHKOW & GRESHAM, PLC  
5727 N. 7<sup>th</sup> Street, Suite 409  
Pheonix, Arizona 85014

CBN/spsr

PATENT APPLICATION FEE DETERMINATION RECORD					Application or Docket Number	
Effective November 10, 1998					09/429357	
<b>CLAIMS AS FILED - PART I</b>					<b>SMALL ENTITY TYPE</b> <input checked="" type="checkbox"/> <b>OR</b> <b>OTHER THAN SMALL ENTITY</b>	
(Column 1)		(Column 2)				
FOR	NUMBER FILED	NUMBER EXTRA		RATE	FEE	
BASIC FEE					380.00	
TOTAL CLAIMS	29 minus 20 = *	9		X\$ 9=	81	
INDEPENDENT CLAIMS	3 minus 3 = *	—		X39=		
MULTIPLE DEPENDENT CLAIM PRESENT				+130=		
				TOTAL	961	
* If the difference in column 1 is less than zero, enter "0" in column 2					OR TOTAL	
<b>CLAIMS AS AMENDED - PART II</b>					<b>SMALL ENTITY</b> <b>OR</b> <b>OTHER THAN SMALL ENTITY</b>	
(Column 1)		(Column 2)		(Column 3)		
AMENDMENT A	CLAIMS REMAINING AFTER AMENDMENT		HIGHEST NUMBER PREVIOUSLY PAID FOR	PRESENT EXTRA		
	Total	* 29	Minus	** 29	= 0	
	Independent	* 3	Minus	*** 3	= —	
	FIRST PRESENTATION OF MULTIPLE DEPENDENT CLAIM					
					<b>SMALL ENTITY</b> <b>OR</b> <b>OTHER THAN SMALL ENTITY</b>	
(Column 1)		(Column 2)		(Column 3)		
AMENDMENT B	CLAIMS REMAINING AFTER AMENDMENT		HIGHEST NUMBER PREVIOUSLY PAID FOR	PRESENT EXTRA		
	Total	* —	Minus	** —	= —	
	Independent	* —	Minus	*** —	= —	
	FIRST PRESENTATION OF MULTIPLE DEPENDENT CLAIM					
					<b>SMALL ENTITY</b> <b>OR</b> <b>OTHER THAN SMALL ENTITY</b>	
(Column 1)		(Column 2)		(Column 3)		
AMENDMENT C	CLAIMS REMAINING AFTER AMENDMENT		HIGHEST NUMBER PREVIOUSLY PAID FOR	PRESENT EXTRA		
	Total	* —	Minus	** —	= —	
	Independent	* —	Minus	*** —	= —	
	FIRST PRESENTATION OF MULTIPLE DEPENDENT CLAIM					
* If the entry in column 1 is less than the entry in column 2, write "0" in column 3.						
** If the "Highest Number Previously Paid For" IN THIS SPACE is less than 20, enter "20."						
*** If the "Highest Number Previously Paid For" IN THIS SPACE is less than 3, enter "3."						
The "Highest Number Previously Paid For" (Total or Independent) is the highest number found in the appropriate box in column 1.						


[Advanced Search](#) [Preferences](#) [Language Tools](#) [Search Tips](#)

Quicktime 4

Google Search

**Web** - [Images](#) - [Groups](#) - [Directory](#) -  
Searched the web for **Quicktime 4**.

Results 1 - 10 of about 831,000. Search took 0.33 seconds.

Categories: [Computers > Multimedia > Music and Audio > Software > Macintosh](#)  
[Computers > Systems > Apple > Macintosh > Development > Languages > HyperCard](#)

### QuickTime Download

... anytime. Watch Now 4. Do it yourself editing. Edit movies with the simplicity of cut, copy, and paste. Watch Now 5. Create slide shows. **QuickTime Pro** enables ...  
 Description: Apple's technology for video, sound, animation, graphics, text, music, and 360 degree virtual reality (VR).  
 Category: [Computers > Multimedia > Music and Audio > Software > Macintosh](#)  
[www.apple.com/quicktime/download/](http://www.apple.com/quicktime/download/) - 35k - [Cached](#) - [Similar pages](#)

### Apple - QuickTime

Apple The Apple Store Switch .Mac **QuickTime** Apple Support Mac OS X. Download  
 Movie Trailers What's On Why **QuickTime** Products Tools & Tips Developer. ...  
 Description: **QuickTime** is Apple's multi-platform, industry-standard, multimedia software architecture  
 Category: [Computers > Systems > Apple > Macintosh > Development > Languages > HyperCard](#)  
[www.apple.com/quicktime/](http://www.apple.com/quicktime/) - 20k - [Cached](#) - [Similar pages](#)  
[\[ More results from www.apple.com \]](#)

### アップル - QuickTime - ダウンロード

... Watch Now 4. 編集機能 カット、コピー ... MacintoshでもWindows PC  
 でも、QuickTime Proの全機能を駆使して高品質なオーディ  
 オ ...  
[www.apple.co.jp/quicktime/download/](http://www.apple.co.jp/quicktime/download/) - 20k - [Cached](#) - [Similar pages](#)

### アップル - QuickTime

Apple The Apple Store Switch .Mac **QuickTime** サポート Mac OS X. ダウ  
 ンロード Movie Trailers **QuickTime** TV Hot Picks 製品 デベロッパ. ...  
 Description: **QuickTime**を使ったストリーミング放送を紹介。  
 Category: [World > Japanese > ニュース > オンライン・メディア](#)  
[www.apple.co.jp/quicktime/](http://www.apple.co.jp/quicktime/) - 12k - [Cached](#) - [Similar pages](#)  
[\[ More results from www.apple.co.jp \]](#)

### Quicktime 4 Linux

... **Quicktime 4** Linux was the first convenient way to read and write uncompressed **Quicktime**  
 movies on Linux. Today **Quicktime 4** Linux is intended for content ...  
[heroinewarrior.com/quicktime.php3](http://heroinewarrior.com/quicktime.php3) - 7k - [Cached](#) - [Similar pages](#)

### Macromedia - Flash and QuickTime 4

... Macromedia Flash offers native support for importing, extending, and  
 exporting **QuickTime 4** movies. Additionally, Apple ... and more! ...  
[www.macromedia.com/software/flash/qt4/](http://www.macromedia.com/software/flash/qt4/) - 12k - [Cached](#) - [Similar pages](#)

### 下載 QuickTime

... 10月13日後所購買的 **QuickTime Pro** 註冊碼可在 **QuickTime 4** 以及 **QuickTime**  
**5** 上使用。在此之前所購買的註冊碼做能在 **QuickTime 4** ...  
[www.apple.com.tw/quicktime/download/](http://www.apple.com.tw/quicktime/download/) - 17k - [Cached](#) - [Similar pages](#)

### 蘋果電腦 - QuickTime

... Spiderman Sade. 看看她2月4日的“Lover's Live”聽樂會。 , This  
 Old House ... Feeder Resident Evil. 看看 **QuickTime** 獨家的電影預告片。 ...  
[www.apple.com.tw/quicktime/](http://www.apple.com.tw/quicktime/) - 13k - [Cached](#) - [Similar pages](#)  
[\[ More results from www.apple.com.tw \]](#)

### Interface Hall of Shame - QuickTime 4.0

... Selecting "About **QuickTime** Player..." from the Help menu causes one "About" dialog  
 to be displayed; the other About Box is accessed by selecting "About ...  
[www.iarchitect.com/qtime.htm](http://www.iarchitect.com/qtime.htm) - 43k - [Cached](#) - [Similar pages](#)

**QuickTime Download**

... Get exclusive access to cool QuickTime content, like high-bandwidth movie trailers, games, music and more. 4. Do it yourself editing. ...  
www.apple.com.au/quicktime/download/ - 27k - [Cached](#) - [Similar pages](#)

Goooooooooooooogle ▶

Result Page:    [1](#) [2](#) [3](#) [4](#) [5](#) [6](#) [7](#) [8](#) [9](#) [10](#)    [Next](#)

Quicktime 4

Google Search

[Search within results](#)

Dissatisfied with your results? [Help us improve.](#)

Try your query on: [AltaVista](#) [Excite](#) [Lycos](#) [Yahoo!](#)

Get the [Google Toolbar](#):



[Google Home](#) - [Advertise with Us](#) - [Search Solutions](#) - [News and Resources](#) - [Language Tools](#) - [Jobs, Press, Cool Stuff...](#)

©2002 Google



Google Search: RealThings

http://www.google.com/search?hl=en&amp;ie=ISO-8859-1&amp;q=RealThings


[Advanced Search](#) [Preferences](#) [Language Tools](#) [Search Tips](#)

RealThings

Google Search

[Web](#) [Images](#) [Groups](#) [Directory](#)
Searched the web for **RealThings**.

Results 1 - 10 of about 2,620. Search took 0.32 seconds.

Did you mean: **Real Things****IBM/Ease of Use/RealThings Design Guide**

... **RealThings** Design Guide **RealThings** is a design approach using counterparts that resemble familiar, real-world artifacts. It ... work. ...

[www-3.ibm.com/ibm/easy/eou\\_ext.nsf/Publish/581](http://www-3.ibm.com/ibm/easy/eou_ext.nsf/Publish/581) - 30k - [Cached](#) - [Similar pages](#)

**IBM/Ease of Use/RealThings Design Guide - Print View**

... site. **RealThings** Design Guide **RealThings** is a design approach using counterparts that resemble familiar, real-world artifacts. It ...

[www-3.ibm.com/ibm/easy/eou\\_ext.nsf/Publish/581PrintView](http://www-3.ibm.com/ibm/easy/eou_ext.nsf/Publish/581PrintView) - 60k - [Cached](#) - [Similar pages](#)

[ [More results from www-3.ibm.com](#) ]

**alphaWorks : RealThings : Download**

... **RealThings**. Choose a file to download and click "Download Now". ... RealThingsGuidelines.zip, 714 KB, **RealThings** guideline documentation. ...

[www.alphaworks.ibm.com/aw.nsf/download/realthings](http://www.alphaworks.ibm.com/aw.nsf/download/realthings) - 23k - [Cached](#) - [Similar pages](#)

**alphaWorks : RealThings**

... **RealThings**, download. Date Posted: September 12, 1997. Update:

October 9, 1998. ... **RealThings** is rated, What is **RealThings**? ...

[www.alphaworks.ibm.com/tech/RealThings](http://www.alphaworks.ibm.com/tech/RealThings) - 31k - [Cached](#) - [Similar pages](#)

[ [More results from www.alphaworks.ibm.com](#) ]

**IBM RealThings**

Home IBM **RealThings**. John Mullaly Advanced Human-Computer Interaction User Interface Architecture & Design IBM Corporation Austin, TX USA [jmullaly@us.ibm.com](mailto:jmullaly@us.ibm.com). ...

[www.acm.org/chapters/chiaustin/events/19981119/real.htm](http://www.acm.org/chapters/chiaustin/events/19981119/real.htm) - 9k - [Cached](#) - [Similar pages](#)

**RealThings**

For information, contact Don Williams at: [williamsdl@earthlink.net](mailto:williamsdl@earthlink.net).

[back to ASHOME Home](#).

[home.earthlink.net/~barbarawill/1RealThings.html](http://home.earthlink.net/~barbarawill/1RealThings.html) - 2k - [Cached](#) - [Similar pages](#)

**RealThings People Said..5**

"I'ma misogynist."

[www.birdpress.com/pages/real/real5.html](http://www.birdpress.com/pages/real/real5.html) - 1k - [Cached](#) - [Similar pages](#)

**RealThings People Said..2**

"I have to think about myself more."

[www.birdpress.com/pages/real/real2.html](http://www.birdpress.com/pages/real/real2.html) - 1k - [Cached](#) - [Similar pages](#)

[ [More results from www.birdpress.com](#) ]

**JARS Automatic Resource Listing**

... JARS Automatic Resource Listing Category: Programming - Other Language: Java. **RealThings** (FC4-79) Computer applications whose interface style uses functional ...

[www.jars.com/classes/jresout.cgi?resource=5301](http://www.jars.com/classes/jresout.cgi?resource=5301) - 16k - [Cached](#) - [Similar pages](#)

**[DaveCentral] - DaveCentral: RealThings**

Google Search: RealThings

http://www.google.com/search?hl=en&ie=ISO-8859-1&q=RealThings

... **RealThings** 1.1 Free. **RealThings** are computer applications whose interface style uses functional representations of real-world objects, such as books, phones ...  
[www.davecentral.com/projects/realthings/?topic=120,121](http://www.davecentral.com/projects/realthings/?topic=120,121) - 43k - [Cached](#) - [Similar pages](#)

Did you mean to search for: **Real Things**

Goooooooooooooogle ►

Result Page: 1 2 3 4 5 6 7 8 9 10 [Next](#)

RealThings

Google Search

[Search within results](#)

Dissatisfied with your results? [Help us improve.](#)

Try your query on: [AltaVista](#) [Excite](#) [Lycos](#) [Yahoo!](#)

[Google Home](#) - [Advertise with Us](#) - [Search Solutions](#) - [News and Resources](#) - [Language Tools](#) -  
[Jobs, Press, Cool Stuff...](#)

©2002 Google

## What are cookies?

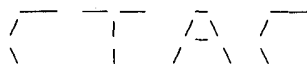
<http://gsbwww.uchicago.edu/computing/cookies.htm>

[ For Public Release ]

-----BEGIN PGP SIGNED MESSAGE-----

---

The U.S. Department of Energy  
Computer Incident Advisory Capability



---

### INFORMATION BULLETIN

#### Internet Cookies

March 12, 1998 23:00 GMT

Number I-034

---

**PROBLEM:** Cookies are short pieces of data used by web servers to help identify web users. The popular concepts and rumors about what a cookie can do has reached almost mystical proportions, frightening users and worrying their managers.

**PLATFORM:** Any platform that can use a modern web browser.

**DAMAGE:** No damage to files or systems. Cookies are only used to identify a web user though they may be used to track a user's browsing habits.

**SOLUTION:** No files are destroyed or compromised by cookies, but if you are concerned about being identified or about having your web browsing traced through the use of a cookie, set your browser to not accept cookies or use one of the new cookie blocking packages. Note that blocking all cookies prevents some online services from working. Also, preventing your browser from accepting cookies does not make you an anonymous user, it just makes it more difficult to track your usage.

---

**VULNERABILITY ASSESSMENT:** The vulnerability of systems to damage or snooping by using web browser cookies is essentially nonexistent. Cookies can only tell a web server if you have been there before and can pass short bits of information (such as a user number) from the web server back to itself the next time you visit. Most cookies last only until you quit your browser and then are destroyed. A second type of cookie known as a persistent cookie has an expiration date and is stored on your disk until that date. A persistent cookie can be used to track a user's browsing habits by identifying him whenever he returns to a site. Information about where you come from and what web pages you visit already exists in a web server's log files and could also be used to track users browsing habits, cookies just make it easier.

---

Internet Cookies

The popular rumors about web cookies describe them as programs that can scan your hard drive and gather information about you including: passwords, credit card numbers, and a list of the software on your computer. None of this is close to the truth. A cookie is a short piece of data, not code, which is sent from a web server to a web browser when that browser visits the server's site. The cookie is stored on the user's machine, but it is not an executable program and cannot do anything to your machine.

Whenever a web browser requests a file from the web server that sent it a cookie, the browser sends a copy of that cookie back to the server along with the request. Thus a server sends you a cookie and you send it back whenever you request another file from the same server. In this way, the server knows you have visited before and can coordinate your access to different pages on its web site. For example, an Internet shopping site uses a cookie to keep track of which shopping basket belongs to you. A server cannot find out your name or e-mail address, or anything about your computer using cookies.

Normally, cookies are only sent back to the server that originally sent them to the browser and to no one else. A server can set the domain attribute for a cookie so that any server in the same Internet subdomain as the computer that sent the cookie will have the cookie sent along with a file request. This is so those larger sites that utilize multiple servers can coordinate their cookies across all the servers. The domain path can not be set to send cookies to a subdomain outside of the subdomain where the server resides.

A cookie is sent to a browser by including a line with the following syntax in the header of an HTML document. Note that the header is removed from the document before the browser displays it. Thus, you will not see the header lines if you execute the View, Source or View, Document Source commands in your browser.

```
Set-Cookie: NAME=VALUE; expires=DATE;path=PATH; domain=DOMAIN_NAME; secure
```

Here the upper case names are strings the server can set.

NAME=VALUE is the name of the cookie and its VALUE. This is the data that the web server wants passed back to it when a browser requests another page.

DATE is an attribute that determines how long the cookie persists on your system. If there is no expiration date, the cookie is stored in memory only and expires at the end of the current session (that is, when you quit the web browser). If the DATE attribute is in the future, the cookie is a persistent cookie and is saved in a file. Only persistent cookies can be used to track a user at more than one site. Setting the date for an existing cookie to be some day in the past deletes the cookie.

DOMAIN\_NAME is an attribute that contains the address of the server that sent the cookie and that will receive a copy of this cookie when the browser requests a file from that server. It defaults to the server that set the cookie if it is not explicitly set in the Set-Cookie: line. DOMAIN\_NAME may be set to equal the subdomain that contains the server so that multiple servers in the same subdomain will receive the cookie from the browser. This allows larger web sites to coordinate multiple servers in the same subdomain. For example, if the DOMAIN\_NAME equals www.mydomain.com then machines named one.www.mydomain.com, two.www.mydomain.com, and three.www.mydomain.com would all receive the cookie from the browser. The value of DOMAIN\_NAME is limited such that only hosts within the indicated subdomain may set a cookie for that

subdomain and the subdomain name is required to contain at least two or three dots in it. Two dots are required if the top level domain is: .COM, .EDU, .NET, .ORG, .GOV, .MIL, or .INT. Three dots are required for any other domain. This is to prevent the subdomain from being set to something like .COM, the subdomain of all commercial machines.

PATH is an attribute that is used to further refine when a cookie is sent back to a server. When the PATH attribute is set, a cookie is only sent back to the server if both the DOMAIN\_NAME and the PATH match for the requested file.

secure is an attribute that specifies that the cookie is only sent if a secure channel (https) is being used.

#### What Information Can A Server Get From A Browser

=====

When a browser sends a request to a server, it includes its IP address, the type of browser you are using, and the operating system of your computer. This information is usually logged in the server's log file. A cookie sent along with the request can add only that information, which is contained in the cookie and which, was originally sent to the browser by the same server. Thus, there is no additional personal information explicitly sent to the server by allowing cookies.

#### Cookies and shopping Sites

=====

As mentioned above, cookies are used by Internet shopping sites to keep track of you and your shopping cart. When you first visit an Internet shopping site, you are sent a cookie containing the name (ID number) of a shopping cart. Each time you select an item to purchase, that item is added to the shopping cart. When you are done with your shopping, the checkout page lists all the items in the shopping cart tied to that cookie. Without cookies, you would have to keep track of all the items you want to buy and type them into the checkout page or buy each item, one at a time.

Another method is for the shopping site to send a separate cookie containing the item number to your browser whenever you select an item to purchase. Your browser sends all those cookies along with the request for the checkout page. The checkout page uses the cookies to make a list of the items you want to purchase.

#### Cookies and Custom Home Pages

=====

Another use of cookies is to create customized home pages. A cookie is sent to your browser for each of the items you expect to see on your custom home page. Whenever you request your custom home page your cookies are sent along with the request to tell the server which items to display. Without cookies, a server would require you to identify yourself each time you visit the custom page so it knows what items to display. The server would also have to store the custom page settings for every visitor.

#### Cookies and Buying Habits

=====

One of the less admirable uses of cookies, and the one that is causing all the controversy, is its use as a device for tracking the browsing and buying habits of individual web users. On a single web site or a group of web sites within a single subdomain, cookies can be used to see what web pages you visit and how often you visit them. This information is also in the server's log files and so the use of a cookie here does not increase a server's ability to track you, it just makes it easier.

On multiple client sites being serviced by a single marketing site, cookies can be used to track your browsing habits on all the client sites. The way this works is a marketing firm contracts with multiple client sites to display its advertising. The client sites simply put an <IMG> tag on their web pages to display the image containing the marketing firm's advertisement. The tag does not point to an image file on the client's machine but contains the URL of the marketing firm's advertisement server and includes the URL of the client's page. Thus when you open a page on the client's site the advertisement you see was actually obtained from the advertising firm's site.

The advertising firm sends a cookie along with the advertisement, and that cookie is sent back to the advertising firm the next time you view any page containing one of its advertisements. If many web sites support the same advertising firm, that firm will be able to track your browsing habits from page to page within all the client sites. They will not be able to see what you do with the pages you view; they will only know which pages you are viewing, how often you view them, and the IP address of your computer. This information can be used to infer the things you are interested in and to target advertising to you based on those inferences.

NOTE: A URL is a Uniform Resource Locator, which is a string containing the type of resource, IP address of the server machine containing the resource, and the path to the resource on the server. When you access a web page, the URL is what you type in the address field of the web browser. For example: <http://ciac.llnl.gov/ciac/CIACHome.html> is a URL for the CIACHome.html document, which is an http document, on the ciac.llnl.gov server in the /ciac directory.

#### Examining Persistent Cookies Already On Your System

=====

Persistent cookies are stored in different places on your system depending on which web browser and browser version you are using. Netscape stores all its persistent cookies in a single file named cookies.txt on the PC or magiccookie on the Macintosh. Both files are in the Netscape directory. You can open and edit this file with a text editor and delete any cookies that you don't want to keep or delete the file itself to get rid of all of your cookies.

Internet Explorer stores persistent cookies in separate files named with the user's name and the domain name of the site that sent the cookie. For example: yourname@ciac.txt. The cookie files are stored in /Windows/cookies or in /Windows/profiles/<username>/cookies directories, where <username> is replaced with the user's login name. If your operating system directory is not named Windows (such as Winnt for Windows NT) then look in that directory instead of the Windows directory. You can delete any of these files you do not want to keep.

You can open these files to see where they came from and what information they contain. For example, the following are the contents of an Internet Explorer

cookie file.

```
Counter_Cookie
7
www.myplace.com/Java/
0
2750889984
29260821
2802449904
29177426
*
```

This particular cookie file was named orvis@java.txt. The file name is made up of the username (orvis) and the last part of the domain (java). The text "Counter\_Cookie" is the name of the cookie and 7 is its value. The URL is the domain attribute and the numbers contain the date and other cookie attributes. This particular cookie implements a page counter that lists how many times you have visited a particular page. Whenever you visit that page, this cookie is sent along with the page request. The server then knows that this is the eighth (7 + 1) time you visited the page and inserts that number into the web page. It then increments the value of the cookie from 7 to 8 and sends it back to the browser along with the requested page. The new cookie replaces the old one so the next time you visit the number 8 is sent to the server. See the example in the "Cookies, VBScript, JavaScript, and Java" section below to see this page in action.

#### Preventing Any Cookies from being Placed On Your System

You can prevent any cookies from being sent to your system using the browser options. In Internet Explorer 4.0, choose the View, Internet Options command, click the Advanced tab and click the Disable All Cookie Use option. In Netscape 4.0, choose the Edit, Options command, click on Advanced and click the Disable Cookies option. After that, no cookies will be stored on your system. You will need to turn cookies back on if you want to use any online services that require them. You can also choose the option to prompt you before accepting a cookie, but at many sites you will be continually closing the warning dialog box.

If you are using earlier versions of Netscape or Internet Explorer, you can require that the browser warn you before accepting a cookie, but it cannot block all cookies. At a busy shopping site, acknowledging all the warnings can get really tedious. There are some other methods for fooling your browser into not accepting a cookie discussed in the cookie web pages listed at the end of this bulletin.

#### Cookie Blocking Software

Several companies are offering special software packages that work with your web browser to control who can send you a cookie. In these packages, you designate which sites can send you a cookie and which can not, alleviating the need to turn cookie use on and off by hand. If you want to use cookies in some instances and not in others, one of these packages may make things easier.

Several packages are listed at the following sites:

<http://www.cookiecentral.com/files.htm>

<http://www.junkbusters.com/ht/en/links.html#nsclean>

#### Cookies, VBScript, JavaScript, and Java

=====

Programs written in VBScript, JavaScript, and Java that are attached to a web page can read and store cookies on your system. The limitations on these cookies are the same as cookies sent to your browser by the server that sent you the program. Cookies created by these programs can only pass information from one page to the next.

The following site demonstrates a page counter using JavaScript.  
<http://www.sna.com/mm Matteo/Java/jscookies.html>

#### More Cookie Information

=====

The following web sites are just a few of the sites that specialize in cookie information.

Yahoo: <http://www.yahoo.com> search for "cookie".

Netscape's cookie specification:  
[http://www.netscape.com/newsref/std/cookie\\_spec.html](http://www.netscape.com/newsref/std/cookie_spec.html)

Netscape's cookie security FAQ  
<http://search.netscape.com/assist/security/faqs/cookies.html>

Cookie Central: <http://www.cookiecentral.com>

Junkbusters: <http://www.junkbusters.com>

---

CIAC, the Computer Incident Advisory Capability, is the computer security incident response team for the U.S. Department of Energy (DOE) and the emergency backup response team for the National Institutes of Health (NIH). CIAC is located at the Lawrence Livermore National Laboratory in Livermore, California. CIAC is also a founding member of FIRST, the Forum of Incident Response and Security Teams, a global organization established to foster cooperation and coordination among computer security teams worldwide.

CIAC services are available to DOE, DOE contractors, and the NIH. CIAC can be contacted at:

Voice: +1 510-422-8193  
FAX: +1 510-423-8002  
STU-III: +1 510-423-2604  
E-mail: [ciac@llnl.gov](mailto:ciac@llnl.gov)

For emergencies and off-hour assistance, DOE, DOE contractor sites, and the NIH may contact CIAC 24-hours a day. During off hours (5PM - 8AM PST), call the CIAC voice number 510-422-8193 and leave a message,



or call 800-759-7243 (800-SKY-PAGE) to send a Sky Page. CIAC has two Sky Page PIN numbers, the primary PIN number, 8550070, is for the CIAC duty person, and the secondary PIN number, 8550074 is for the CIAC Project Leader.

Previous CIAC notices, anti-virus software, and other information are available from the CIAC Computer Security Archive.

World Wide Web: <http://www.ciac.org/>  
(or <http://ciac.llnl.gov> -- they're the same machine)  
Anonymous FTP: <ftp.ciac.org>  
(or [ciac.llnl.gov](http://ciac.llnl.gov) -- they're the same machine)  
Modem access: +1 (510) 423-4753 (28.8K baud)  
+1 (510) 423-3331 (28.8K baud)

CIAC has several self-subscribing mailing lists for electronic publications:

1. CIAC-BULLETIN for Advisories, highest priority - time critical information and Bulletins, important computer security information;
2. SPI-ANNOUNCE for official news about Security Profile Inspector (SPI) software updates, new features, distribution and availability;
3. SPI-NOTES, for discussion of problems and solutions regarding the use of SPI products.

Our mailing lists are managed by a public domain software package called Majordomo, which ignores E-mail header subject lines. To subscribe (add yourself) to one of our mailing lists, send the following request as the E-mail message body, substituting ciac-bulletin, spi-announce OR spi-notes for list-name:

E-mail to [ciac-listproc@llnl.gov](mailto:ciac-listproc@llnl.gov) or [majordomo@tholia.llnl.gov](mailto:majordomo@tholia.llnl.gov):  
subscribe list-name  
e.g., subscribe ciac-bulletin

You will receive an acknowledgment email immediately with a confirmation that you will need to mail back to the addresses above, as per the instructions in the email. This is a partial protection to make sure you are really the one who asked to be signed up for the list in question.

If you include the word 'help' in the body of an email to the above address, it will also send back an information file on how to subscribe/unsubscribe, get past issues of CIAC bulletins via email, etc.

PLEASE NOTE: Many users outside of the DOE, ESnet, and NIH computing communities receive CIAC bulletins. If you are not part of these communities, please contact your agency's response team to report incidents. Your agency's team will coordinate with CIAC. The Forum of Incident Response and Security Teams (FIRST) is a world-wide organization. A list of FIRST member organizations and their constituencies can be obtained via WWW at <http://www.first.org/>.

This document was prepared as an account of work sponsored by an agency of the United States Government. Neither the United States Government nor the University of California nor any of their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or

usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial products, process, or service by trade name, trademark, manufacturer, or otherwise, does not necessarily constitute or imply its endorsement, recommendation or favoring by the United States Government or the University of California. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or the University of California, and shall not be used for advertising or product endorsement purposes.

LAST 10 CIAC BULLETINS ISSUED (Previous bulletins available from CIAC)

I-024: CGI Security Hole in EWS1.1 Vulnerability  
 I-025A: Windows NT based Web Servers File Access Vulnerability  
 I-026: Vulnerability in ssh-agent  
 I-027B: HP-UX Vulnerabilities (CUE, CDE, land)  
 I-028: Vulnerabilities in CDE  
 I-029: IBM AIX Telnet Denial-of-Service Vulnerability  
 I-030: SunOS volrmmount (1) Vulnerability  
 I-031A: WindowsNT-95 Attacks on DOE Sites  
 I-032: Sun Solaris Vulnerabilities (vacation, dtaction)  
 I-033: Sun Solaris Vulnerabilities (nnd, rpc.cmsd)

-----BEGIN PGP SIGNATURE-----

Version: 4.0 Business Edition

iQCVAwUBNQh8oLnzJzdsy3QZAQG2hQP8CQk3IM0FpjgGs0sxK+LTNPiD8lUMxVMY  
 NTPsYBeOIakiF0KWga/Lrc/ai6gCsha6jKvDYHKAAU4t/bevd46jtCelit3oBysT  
 5ps08tG7iokLOM4NOINZoxoP8PN/uY0Tm3PzUCSwfAQwe6eQ85nipDoB1sU5bcdo  
 NX5HFL4QY9w=  
 =y1Fl

-----END PGP SIGNATURE-----

Return to the [virus page](#) or to [GSB Computing Services](#)

RealThings Design Guide

RealThings is a design approach using counterparts that resemble familiar, real-world artifacts. It is based on the premise that computing can fulfill functional and productive goals, while also being approachable, fun, and visually compelling for novice and casual users. This discussion contains the premise of the approach, prototypical examples demonstrating the approach as an alternative to the graphical user interface (GUI) style typically in use, guidelines and principles for applying the approach, and a white paper introducing the motivation for exploring this approach.

**Premises** Benefits of the RealThings approach.

**Examples** RealPhone, RealCD, and RealBook as prototypical examples of this approach.

**Guidelines** Issues to consider in applying the approach to a real design.

**White Paper** More detailed discussion of the advantages of RealThings over WIMP (windows, icons, menus and pointer) interfaces.

**The Team** Contributors to the development of this work.

Premises

Focus

Today's computer-based user interface mechanisms can be removed and replaced by more natural and intuitive ones allowing designers and users to focus on information content.

A typical word processor can consume up to 30% of the display with title bars, menu bars, tool bars, and status lines. Even though the RealThings we've developed so far don't match a word processor's high level of function, we do believe that multiple design strategies can give most of the space back to the user while simplifying user interaction at the same time.

For example, the RealBook™ removes the typical GUI visual clutter providing more space for content and more intuitive interactions for the user. The natural use of covers, tabs, and pages enables the user to focus on the task.

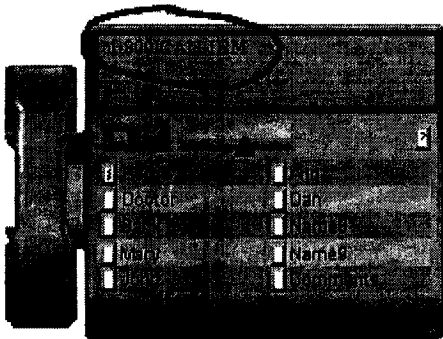
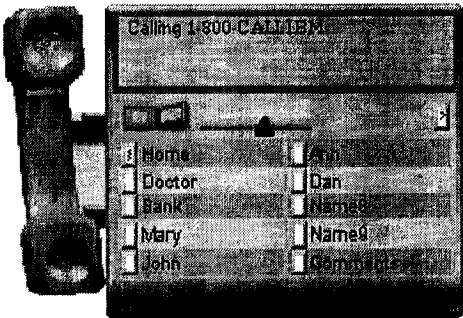
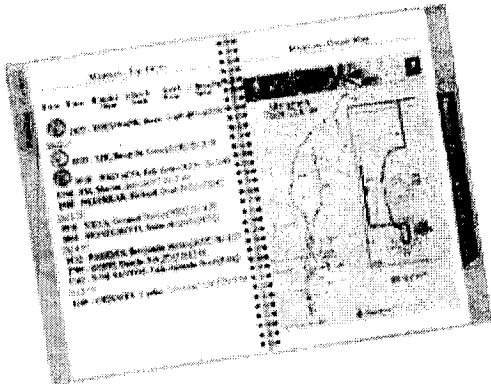
Transition

Simulation of real-world interaction mechanisms can provide a bridge into the computer realm for new and novice users. Some techniques may even be applicable for more advanced users, especially when the techniques are so familiar they are essentially transparent to the task.

For example, the RealPhone™ provides a handset similar to those in the real world. Clicking the handset causes it to come off-hook answering the phone or placing a call. Obviously, a handset isn't necessary on a computer-based phone - most computer phones simply use a push button labeled Dial. However, we found that many users intuitively understand the role of the handset without needing any instructions or labels. The RealPhone also provides an animated, slideout drawer containing speed-dial numbers and other settings. Tests have shown that users enjoy the interaction and appreciate the reduced clutter.

Value-add

Using real-world objects as a starting point, we can add significant value over real-world objects and mechanisms through the use of the computer in a complimentary manner.



The current implementation of the RealPhone was designed primarily to explore the Focus and Transition premises; however, it does enhance the real-world phone by allowing the user to dial an alphanumeric phone number. The user simply types the characters, such as 1-800-CALLIBM. While this is just one minor enhancement, imagine what might be accomplished with further analysis and design.

For example, the myriad of options available on telephones today, such as call waiting, call forwarding, caller-id, caller-id blocking, and many more, have created a user model so complicated that most people don't even know there are exclusive interactions between these functions - some must be cancelled in order to enable others. The interaction using today's real-world telephone requires dialing a series of \*nn codes such as dialing \*70 to cancel call waiting. This is an area of great opportunity for telephone implementations on the computer. An implementation such as the RealPhone, extended to provide this added value, might even be a model on which future real-world (computerized) telephones could be designed.

of its own - limits that we can surpass by using computers.

Presentation

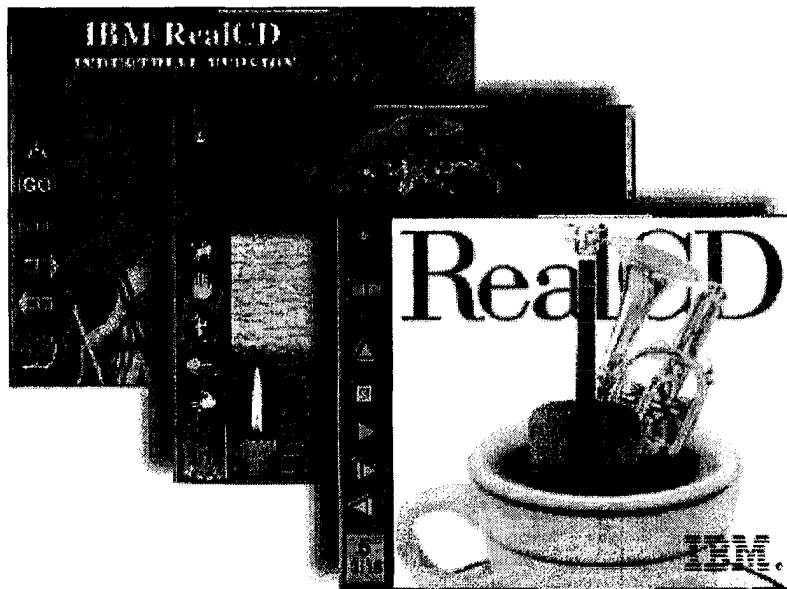
Strictly emulating the real-world may not be the best answer for all kinds of objects and all tasks. While it may intrigue and delight new users, and entice them to delve deeper in exploration, the real-world imposes limits

A realistic design for high-function applications may require a combination of presentations ranging from realistic for common and simple tasks to abstract for complex ones.

Some tasks are by nature abstract and have no obvious real-world representation. It is our expectation that some of the interaction mechanisms developed for RealThings will be appropriate and useful even for these tasks.

One approach to addressing a spectrum of objects, task requirements, and users is to base implementations on a *model-view paradigm*. The model-view paradigm separates object semantics, *what* an object does, from its view, which establishes *how* the user interacts with it. In fact, we have experimented with extending this paradigm to model-view-*presentation*, which makes a further distinction between the abstract definition of the view and its actual rendering.

Both approaches allow developers to create objects that can present different appearances and interaction techniques to users. This means each user can have an interface style that is appropriate, and that they are most comfortable using, in the current situation. The RealCDs™ shown below present three different presentation styles.



We are also designing programming frameworks that make it easy and productive for developers to provide multiple presentation and interaction styles for an object. This approach will allow third parties to provide alternative styles, giving more choices to users - similar to the multiple screen savers and color-scheme options offered today.

## Examples

At IBM we believe that your experience with computer programs does not have to be boring nor does it have to lead you through a series of non-intuitive steps to get things done. User Interface Architects in IBM's Ease of Use organization have worked on exploring user interfaces that transfer users' knowledge of how things work in the real world and puts that knowledge into software applications that perform the same function. The following are three such examples:

### RealPhone

The RealPhone is a speaker-phone that supports dialing and answering, a name/number list, 10 speed dial buttons, flash, mute, and redial. You can use either a mouse or the keyboard for all interactions. It was implemented using the non-rectangular window capability of Windows 95.

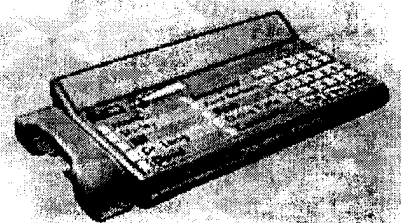
The RealPhone was the first RealThing we designed, and much of what we learned was incorporated into the RealCD and the RealThings Guidelines. We have not revisited the RealPhone to include aspects of the guidelines that were added later.

### RealCD



The RealCD interface combines a familiar real-world object with content and controls for interacting with that content. The CD case is easily recognized and understood, and it emphasizes the content rather than the interface. Through careful design, we can retain those positive aspects of real-world objects in software, while augmenting them with the power of the computer.

The RealCD plays audio CD's. Its interface resembles a plastic CD jewel case with a control panel for the basic playing functions. The case can be opened to display the CD booklet and playlist. Several pictures are included for the case and booklet covers; however, users can also incorporate their own scanned or drawn pictures. Users enter the CD title and playlist (track titles). These customizations appear whenever the CD is played.

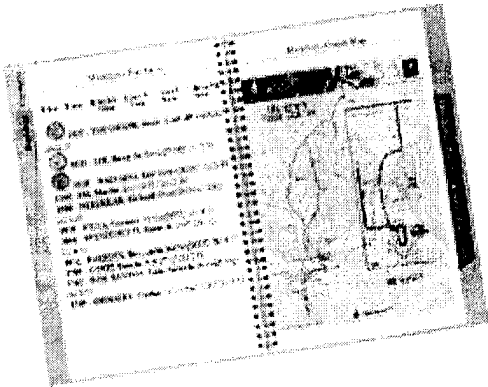


The RealCD was implemented in Java.

**RealBook**

The RealBook is an information container that simulates the operation of a book. The book was implemented in Java and can be populated with three categories of content: covers, tabbed separators, and documents. Documents can contain HTML or Java AWT components, and the content can flow across several pages within a tabbed section.

Single and facing page views, as well as left to right and top to bottom orientations are also supported. A simple Java interface enables developers to include information in the book.



**Guidelines**

A major premise of RealThings is that users should be able to recognize an object and predict its behavior based on their experiences with the object's real-world counterpart.

**Use naturally occurring shapes for RealThings to aid recognition by users**



Use non-rectangular windows when the user environment supports them. In Windows 95 a window's shape can be defined by the use of a non-rectangular region. Use of non-rectangular windows can significantly enhance the real-world effect of RealThings. It can also provide better screen space utilization. The RealPhone uses a non-rectangular window.

When non-rectangular windows are not supported, transparency may be used to assist in creating a similar visual effect. Objects will appear more natural and can be more quickly identified when they have characteristic shapes that are easily recognized.

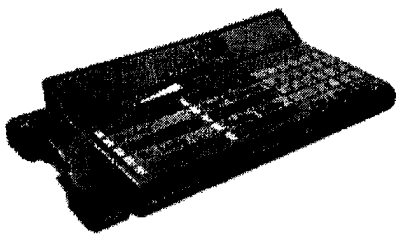
**Use natural and intuitive affordances**

Provide natural affordances as interaction mechanisms wherever possible. Handles and other grasping surfaces provide immediate visual cues that most users will recognize. Keep the interactions simple, such as a single mouse click, and avoid real world controls that are difficult to emulate using the input devices provided. For example, rotating controls such as knobs and dials are difficult to map to the x/y movement of a pointing device such as a mouse.

The Realphone uses a thumb indentation and arrows to indicate the tray handle. A simple click on the handle slides the tray out providing accesses to the notebook.

**Generate Images from 3D models to provide a compelling visual experience**

RealThings are created for today's two-and-a-half dimension desktops but to achieve realism we use bitmap images

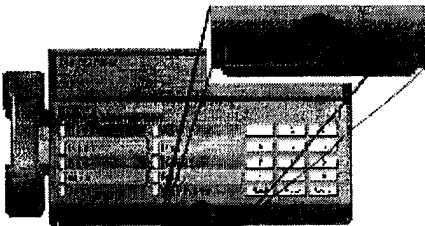


generated from 3D models. Three-dimensional modeling tools typically provide the ability to control camera orientation, to capture bitmaps from different angles, and can provide impressive lighting effects with reflections and shading accurately rendered. The best part is that rendering isn't real-time, it's done during development so you can spend as much time as you want creating a truly compelling image.

The bitmaps should be taken with a camera nearly perpendicular to the surface you want to render so that controls such as buttons, sliders, and entry fields will look natural when mapped onto the surfaces. For the RealPhone we used an angle of about ten degrees so the user can clearly see the edges of the tray that slides out from the front of the phone.

As processor speed and our ability to render 3D objects in real time increases we will be able to render the 3D models in real-time and permit alternate viewing points.

**Use sound to provide cues and feedback, and to create the desired ambiance**



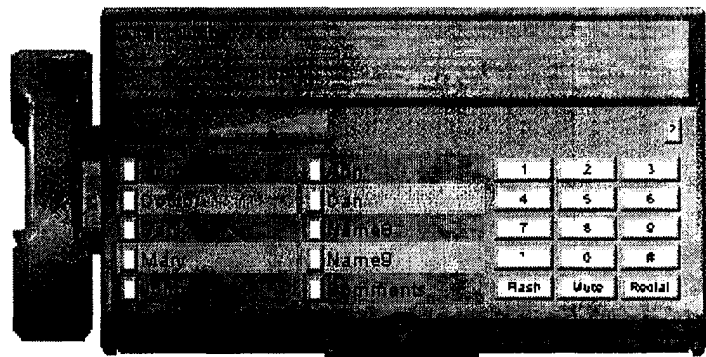


The real world is full of sounds that create ambiance and provide us with cues and feedback. RealThings should provide user options for sound support, including the ability to turn sounds off, and select levels of sound support. Sounds should be consistent with their real-world equivalents to ensure that they correspond to the user's expectations.

We suggest separating sounds into three types: background or ambiance creating sounds, such as a ticking sound for a clock; normal function sounds, such as a click when a button is pushed; and alarm sounds when events occur that require the user's attention, such as a cuckoo sound for a clock, or a bell for a ringing telephone.

**Use animation to convey relationships**

Many people are captivated by animation effects, especially in today's relatively static interfaces. But animation has a important functional role as well. It provides a way of showing relationships between components, such as the tray that slides out from the RealPhone.



The relationship between the tray and the phone is unambiguous. It's clear that the tray is contained within the phone. It doesn't just suddenly appear and does not exist independent of the phone.

It's important not to over use animation. Too many animations at the same time or an animation that continues on longer than useful becomes an annoyance.

**Guidelines: Value-Add**

Extend real-world functionality by incorporating computers' capabilities. Don't constrain the design to include only those functions available in the real-world counterpart.

**Disclose discrete levels of function incrementally**

Economic use of screen space along with incremental introduction of function are two complementary features demonstrated by RealThings. The use of real-world metaphors might lead you to believe that RealThings would be wasteful of valuable screen space. After all, real-world objects are often encased in lots of plastic, steel, or wood that has no function other than physically containing the object's "contents".

In the computer interface our goal is to make each object immediately recognizable by the user. This can be achieved through shape recognition and the use of other visual cues. But we don't need to keep the recognizable version of the object around all of the time. Similarly, the user probably doesn't always need the full-function of an object at their fingertips. Each RealThing provides a sequence of views that use screen space judiciously and present function in logical increments.

**Object views**

In the RealThings implemented so far, we have identified four levels of object views:

- Minimal View
- Base View
- Full-Function View
- Ancillary Views

Each RealThing should provide several of these views but may not need to provide all of them.

**Minimal view**



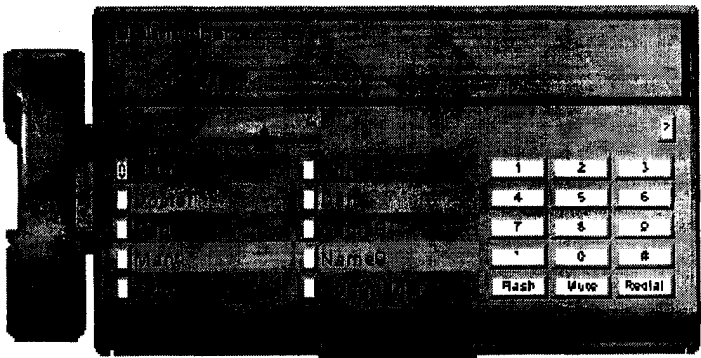
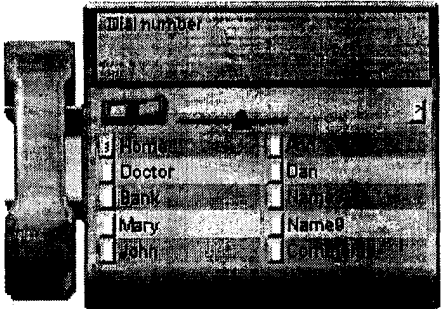
The minimal view provides the most basic level of function and doesn't necessarily resemble the object's real world counterpart. The minimal view of the RealCD consists of only the playing controls (due to Java implications, this view is not in the first release). The RealPhone does not provide a minimal view.

**Base view**

The base view provides a basic level of function in a form that's immediately recognizable by the user. The base view of the RealCD shows the playing controls on the edge of a CD jewel case, which identifies the CD. The RealPhone's base view looks like a real-world telephone. It provides ten speed-dial buttons, a volume control, a functional handset, and a rocker switch that toggles to the full-function view.

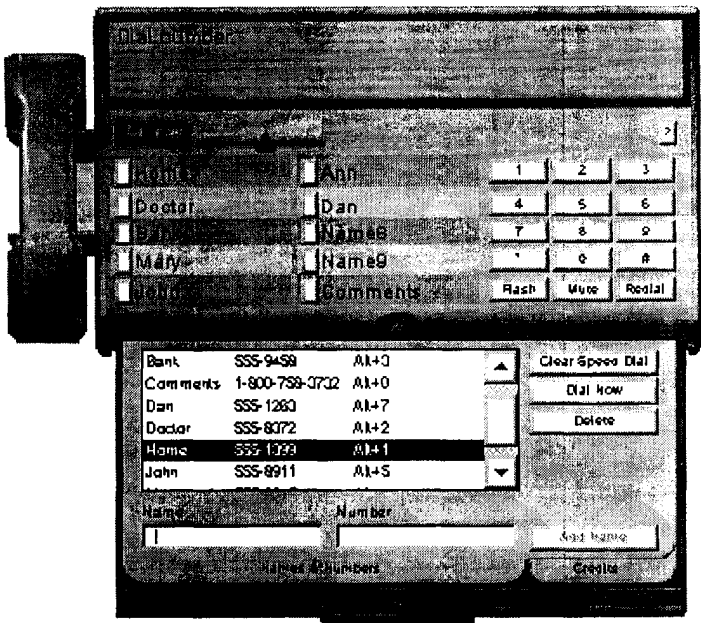
**Full-function view**

The full-function view expands on the base view by adding extra function that may be used less frequently. The full-function view of the RealCD shows an open jewel case that lists the CD tracks. It also makes the CD booklet available. The RealPhone's full-function view, as shown below, adds a numeric pad and buttons for Flash, Mute, and Redial.



**Ancillary views**

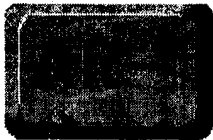
The ancillary views show aspects of the object that may be used occasionally, such as configuration settings and instructions for use. The RealPhone provides an animated slideout tray that contains a notebook. The notebook contains a phone number list, fields for adding new names and numbers, and buttons for dialing a number from the list or assigning a phone number to a speed dial button.



**Component Device Model**

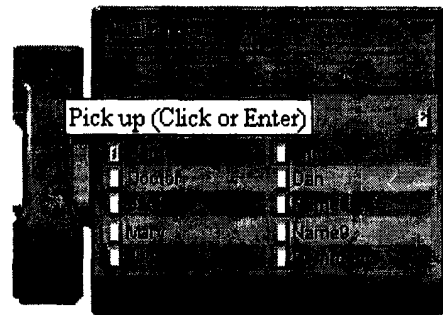
The Component Device Model is a general model that defines a base view, full-function view, slideout tray, and tabbed notebook. It is applicable for implementing a variety of device objects, such as telephones, fax machines, and printers. The RealPhone is an implementation of this model. It's an ideal candidate for implementation as a programming framework. We haven't gotten to that task yet, but if you want to give it a try we would love to talk with you.

**Anticipate and assist frequently occurring tasks**



Provide shortcuts for tasks that users do frequently. For example, the RealPhone makes it very easy for the user to answer when the phone rings, regardless of what the user is doing at the time. When the phone detects an incoming call it becomes the topmost window, but it doesn't take the keyboard focus. The user can keep on working in their current application, or decide to answer to the phone. To answer the phone all the user has to do is give the RealPhone focus. In Windows 95 you can give an object the focus by clicking on it, by using Alt+Tab, or by using a shortcut key. The phone is designed such that when it receives the focus while detecting an incoming call it automatically answers.

**Provide multiple levels of user assistance**



Provide several levels of assistance appropriate to the complexity of the object. Simple cues and reminders can be provided through hover help balloons that appear near the pointer when it pauses over a control for a brief period. The RealPhone uses hover help to identify the function of the control and to tell the user the keyboard equivalent. A pointer-based Help tool can also be used to provide this first level of assistance.

An approach resembling cue cards can be provided for the next level of explanation which should address the role of the control or object in accomplishing tasks.

Cue cards can be extendable to address to the third level which is *how to* information. This information should provide step-by-step instructions for a typical case along with an example.

The highest level of assistance is conceptual. Here you should provide an overview of the object's role, functions, and relationships with other objects. A section in the object's Properties Notebook is an appropriate place for this material.

Provide hints and tips to help users achieve efficient and productive usage quickly. The Properties Notebook is a good place to provide this material also. The contents of Properties Notebooks should also be presented in real-world form, not in traditional GUI form.

From each level of assistance, getting to the next level should be obvious and easy. For example, the cue card might have a button that expands it to include how-to information, and another button that displays the Properties Notebook opened to the concepts section.

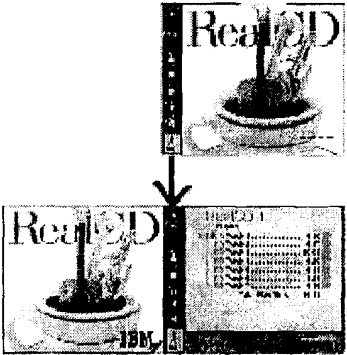
**Keep primary controls in the same position as the user switches between views**

When multiple views of an object are provided, as described in the guideline on Incremental and Discrete Function Disclosure, keep the primary controls in the same position on the screen as the user switches between the views. For example, the RealCD provides a base function view and a full-function view. The player controls remain in the same position as the user switches between the two views.

Also, make it easy for the user to move a view back onto the screen if a portion of it appears off of the screen. If the RealCD's base function view is positioned on the left side of the screen a portion of the full-function view may appear off of the screen. However, the user can easily drag the view back onto the screen. Do not move the view in an attempt to keep it on the screen. Moving the view would cause the controls to move, making it more difficult for the user to work with the object.

**Guidelines: Compatibility**

Currently, RealThings must be usable in the context of a GUI desktop, side-by-side with standard GUI windows. Therefore, they should be designed to be compatible with these standard interfaces.



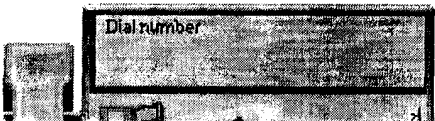
**Adopt standard UI controls and use them consistently**

We recommend the use of several standard user interface elements and techniques. This will not only make objects more recognizable by users, it will allow developers to build RealThings from standardized toolkits. Some of the elements we recommend you standardize are:

- Controls such as buttons, selectors, and fields
- A slideout drawer
- A tabbed notebook

We have identified standard classes of control elements for *activators* and *selectors*.

**Activators** simply cause an action to happen and have no persistent state. They are typically presented as buttons, such as the speed-dial buttons shown on the Realphone to the right.

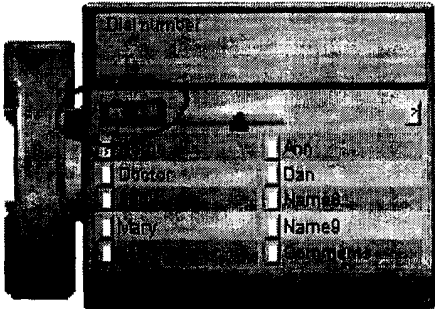




**Selectors** allow the user to pick one or more choices from a set of choices. They are typically presented as switches, dials, sliders, and lists. The RealPhone uses a 1-of-2 selector in the form of a rocker switch to switch between base and full-function views.

These controls should look real and natural, and not be the standard controls provided by current GUI toolkits.

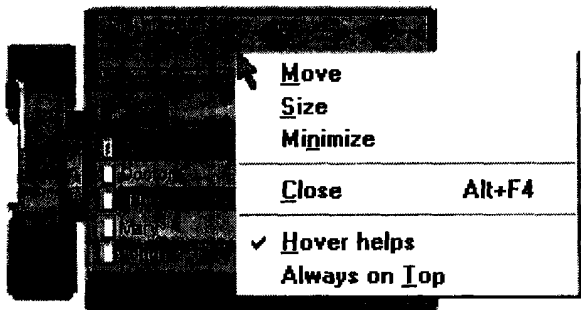
By using a model-view-presentation paradigm in the implementation of these controls the developer can create an appearance that is most appropriate for the real thing being created without having to modify the program's logic, which is encapsulated in the model. Even more interesting, the developer might make several visual styles available for users to choose from.



**Coexist with other desktop applications where it benefits the user**

RealThings run on existing desktops along with native applications. For the most part they use replacement approaches for traditional window borders, titles, borders, and menu bars. But users might expect them to behave in some ways like other applications. For example, Windows 95 applications provide a pop-up menu. For the

RealPhone we chose to provide a pop-up menu to support moving and sizing using the keyboard, and for Close. We also added a Always on Top choice, and supported minimize for users who want to use the Windows 95 Task Bar. We considered using a visual control for Close, something like a power switch, but we decided to use the standard desktop pop-menu in this case.



The remaining RealThings are written in Java so we decided to avoid using platform specific interfaces. Some interfaces, like cut and paste and drag/drop, are fairly common across platforms and should be considered so that RealThings can interact in useful ways with other objects. We plan to explore drag/drop between RealThings and between RealThings and other objects in the future.

**Avoid changing the pointer, except to indicate the current tool**

Use pointer changes sparingly. A constantly changing pointer is distracting and may actually interfere with explicit pointer changes caused by the user. Pointer-based tools, such as found in many drawing programs, are a good way to provide a large number of different functions to the user. The trend is to provide common functions that can be applied across a variety of objects, such as a coloring tool that might be used to color text, bar graphs, buttons, or most any aspect of the interface. If an individual object changes the pointer it may be taking control away from the user and interfering with the user's intent.

**White Paper**

by Justin Richards

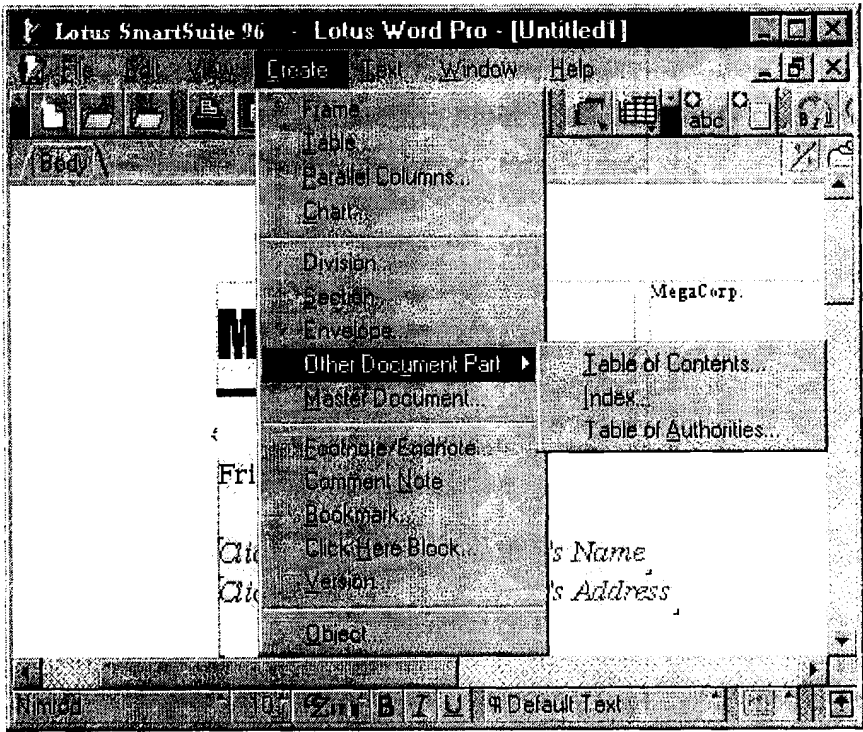
**Today's GUI - The 'WIMP' Interface**

The user interfaces of today are dominated by the so-called *WIMP* UI - Windows, Icons, Menus, Pointers. While there is no denying the success of these interfaces in bringing desktop computing to millions of users across the world, the GUI has grown to be a cluttered, discordant world of clashing icons and wasted screen space.

In the WIMP world, objects (or more usually, applications) are presented in rectangular windows. They do not look real, or even bear more than an occasional passing resemblance to anything in our real world outside the computer. And amongst the visual noise and clutter, are hidden the clues necessary to make the cognitive leap to accommodate a metaphor which relies on the idea that 'windows' can exist on a 'desktop.'

Objects and applications alike are represented by icons. But these icons only show a gross level of information - they indicate the class of object, but rarely impart status information or make important properties apparent. All icons are the same relative size, and pretty much the same shape. And icons must be 'opened' before they are 'alive' enough to do very much.

In the WIMP world, menu bars and tool bars proliferate. Users spend much of their time 'mining' menus or hovering over buttons waiting for help.



As the user fumbles through the interface in search of particular functions, the clutter and visual pollution detracts from the content or task which should be the user's main concern. The UI itself distracts and intimidates the user rather than helping.

Today, the focus of the user interface is often on the GUI controls.

**An Alternative - A Clean Screen UI and RealThings**

In many respects, the way forward for today's content-centric world is obvious. Cut down on the visual clutter and pollution and focus instead on the content. This will have the bonus effect of making that content seem more like 'real world' content as traditional design methodologies and experience can be brought to bear. An escape from the constraints of the rectangular world of the WIMP interface can only bring the UI closer to the world that the user already knows and understands.

In fact, we can already see the start of a reaction against the cluttered WIMP world. Lotus SmartSuite, for example, provided tool bars according to context, rather than present every conceivable function for every possible circumstance.

More interesting is the UI demonstrated by Lotus Organizer, where most operations can be performed in place without resorting to menus or tool bars. Users are presented with a UI that resembles something they already understand - a book. The Smart Center drawers for Calendar (shown here) and Address are also a natural and very usable extension of this philosophy. They are unobtrusive yet accessible - combining the value of pull-down menus with real objects that provide valuable, but not over-comprehensive, function for standard and frequent tasks to the user.

**The World Wide Web**

If 'suite-bloat' is one driver towards a clearer, more accessible world for the users, the World Wide Web is another. The Web is content-based, and that content assumes only the relatively limited built-in function of a Browser. Any other functionality has to be provided intrinsic to the content. Often this function takes the form of pseudo-menus and pick-lists, but even these exhibit a freedom and a freshness of design. This is partly because of the fact that the content is freed from the software and mental constraints of the standard UI tools and widgets provided by the underlying platform.

It is also partly due to the democratization of content. The pages of content on the Web are often put together by designers and novices with little or no previous computer design or programming experience. But, in fact, these are people who are psychologically more able to communicate with the 'common user'. Their designs are not polluted by the design constraints that the UI designer used to have to deal with.

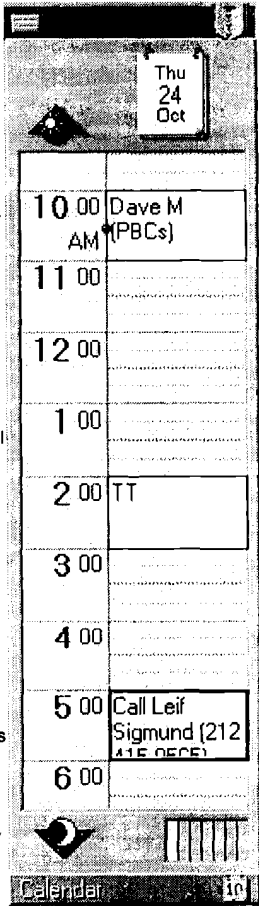
The most interesting design emerging from the Web, though, is where function is intrinsic to content - where objects (for want of a better term) exhibit exactly the behavior that their visual appearance suggests; where function is in support of task; where form suggests purpose; where interaction is apparent from visual appearance.

**Web to WIMPs** It is hardly surprising that there is a general move to incorporate web-like interfaces in the WIMP world. But generally, this is a patchy solution that combines two fundamentally incompatible styles of UI.

**RealThings: Focus on content, not control**

While the general movement is in the right direction, the progression is still quite slow. We can take these notions further still, and the key to this is IBM RealThings.

RealThings exhibit a new, real-world user interface style. They set a new direction in making user interfaces more approachable for novice and casual users. At its simplest, the RealThings philosophy is to make software constructs and applications appear as they would if they existed in the real world outside the computer. Objects and mechanisms are presented in context, and actions are surfaced in more natural ways.

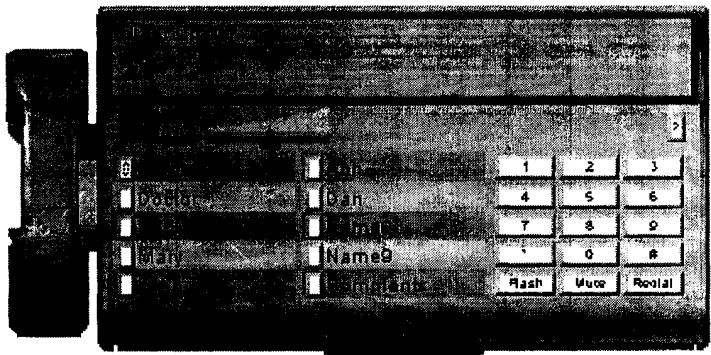


The key to making RealThings work as the next major stage in the development of ease of use is *knowledge transfer from the real world*. Computer users, no matter how new to the interface, no matter how naive, possess a huge body of experience from, and a good understanding of, the real world. By making the behavior of computer objects more natural - more like the user expects from his or her experience of the world - we make it more accessible, less daunting, easier to comprehend and to use. In short, rather than expecting the user to learn what is in effect a new language, we leverage what the user already knows.

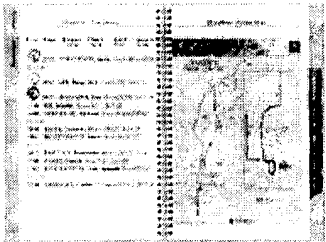
And in doing so, we shift the focus away from controls and back on to content.

**RealThings**

As we have said, the primary motivation for making objects seem more 'real' is to capitalize on what the user already knows and leverage knowledge transfer from the real world. It is not to slavishly copy objects, techniques and interactions from the real world. One example of a RealThing we have created is the IBM RealPhone.



Past development projects include: **RealBook**. Here is an object which will be integral to the user's computer experience - not peripheral and related to just a small subset of tasks. And we are capitalizing on the user's knowledge of real books as well as a design that is well-understood and tested by time.



But we aren't just copying the real-world with its inherent limitations, we are providing navigation techniques that enhance the real world, as well as providing mechanisms and interaction cues that are immediately recognizable and understood.




The **RealCD** will soon be available as another RealThing. In the real world, the user has to take the CD to a player (connected to an amp, speakers, etc), insert it, and only then can they play the CD. Using the RealCD this problem is addressed by bundling the player and the content together - a solution that is impractical in the real world but which addresses the user's actual task.

Having decided to bundle the two, though, the designers have made the CD look like what users could reasonably expect an amalgam of CD and Player to look like *if it did exist* in the real world. The object itself - the composite - is not 'real' but its components are. The CD case looks like a real CD case and provides the visual information about medium and content that is important in the real world. The controls look like the control panel of a CD Player. Given that users have an understanding of the task involved and have knowledge of how that task is performed in the real world, of the objects with which they would interact, this design is easily accessible.

**The Team**

The following are some of the folks that have participated in RealThings design projects:

Concept & Direction	Dick Berry, Roland Merrick
Executive Producer	Tony Temple
Producer	Paul Waldo
Programming	Scott Morgan, Andy Smith, Craig Swearingen
Usability	Scott Isensee, Greg Lubel, Shirley Martin
User Interface Design	John Mullaly, Dave Roberts
Visual Design	Didier Bardon, Martin Morgan
Web Design	Denise Burton, Valerie Fox, Trish Kerr, Linda Lisle



IBM home | Products & services | Support & downloads | My account

Select a country

developerWorks

alphaWorks *emerging technologies*

Licensing

Discussion Forums

alphaWorks Newsletters

Emerging Technologies

☐ All

☐ Java

☐ XML

☐ Web Services

☐ Wireless

Collaboration

more...

find a technology

Robocode Rumble

LEARN TO CODE

Test Your Java Skills

Robocode

Newsletters

Subscribe to the aW flash newsletter:

☒ Text ☐ HTML

[More newsletters](#)

[alphaWorks](#) > [Collaboration](#) > [RealThings](#) > Overview

# RealThings

Date Posted: September 12, 1997

Update: October 9, 1998

The 1-800 number has been removed from the speed dial list.

RealThings is rated **JARS TOP 25%**

## What is RealThings?

RealThings are computer applications whose interface style uses functional representations of real-world objects, such as books, phones, and CDs. Seeing familiar objects makes the transition into the computer realm smoother for new users and can make tasks easier for advanced users as well. RealThings use realistic images, animation, and sound to create a natural and intuitive interface. Books open to pages and tabs, CDs open and play, and phones ring and contain a pull-out tray with a phone number list.

## How does it work?

RealThings provide many features for customization and easy use. Alternative appearances and interaction styles can be chose for each object; in addition, each RealThing has a sequence of views. Each view uses screen space economically and presents functions in logical increments. Also, shortcuts are provided for tasks that are done frequently. For example, when the RealPhone rings, it appears on the screen, without interrupting work in another application.

Platforms: Windows 95

Categories: Collaboration - Media

## RealThings Discussion

[Post New Topic](#) • [Search Forum](#)

[Wicked Interface! Where's CDDDB?](#) by Mickey Mouse on 10/22/99 11:33:40 AM  
.. [I think it died...](#) by Chaoticmass on 02/19/2002 07:08:19 AM

[...more Postings](#)

## About this technol

- Overview
- Requirements
- FAQs
- Download!
- Discussion Forum
- About the Authors
- Evaluate this Technol
- Review Evaluations fr

**Development Reso**  
→ Lotus Software fro  
→ Notes Net

**New Licenseable Tech**  
→ TSpaces  
→ Self Voicing Kit  
→ IRC Client for Java

**Top Online Dem**  
→ Web Services Too  
→ Data Descriptors b  
→ Example

**Top Download**  
→ IRC Client for Java  
→ Speech for Java  
→ NotesBuddy  
→ J323 Engine  
→ Phone for Java

**Poll**  
Do you use peer-to-p  
☐ Yes  
☐ No  
[Vote](#)  
[Poll results](#) [Previous](#)



alphaWorks : RealThings

wysiwyg://10/http://www.alphaworks.ibm.com/tech/RealThings





US005845075A

**United States Patent** [19][11] **Patent Number:** **5,845,075****Uhler et al.**[45] **Date of Patent:** **Dec. 1, 1998**

[54] **METHOD AND APPARATUS FOR DYNAMICALLY ADDING FUNCTIONALITY TO A SET OF INSTRUCTIONS FOR PROCESSING A WEB DOCUMENT BASED ON INFORMATION CONTAINED IN THE WEB DOCUMENT**

[75] Inventors: **Steve Uhler**, Los Altos; **Brent B. Welch**, Mountain View, both of Calif.

[73] Assignee: **Sun Microsystems, Inc.**, Mountain View, Calif.

[21] Appl. No.: **673,956**

[22] Filed: **Jul. 1, 1996**

[51] **Int. Cl.**<sup>6</sup> ..... **G06F 17/30**

[52] **U.S. Cl.** ..... **395/200.3**; 395/200.33; 395/200.48; 707/10; 707/500; 707/513

[58] **Field of Search** ..... 395/200, 500, 395/793, 762, 774, 200.3–200.33, 200.36, 200.48, 200.49, 200.57, 200.68, 680–685; 380/4, 9, 49; 707/513, 500, 10

[56] **References Cited**

#### U.S. PATENT DOCUMENTS

5,530,852	6/1996	Meske, Jr. et al.	395/200.36
5,572,643	11/1996	Judson	395/200.48
5,706,502	1/1998	Foley et al.	395/682
5,706,507	1/1998	Schloss	395/615
5,708,709	1/1998	Rose	380/4
5,710,918	1/1998	Lagarde et al.	395/680
5,745,909	4/1998	Perlman et al.	707/513
5,764,916	6/1998	Busey et al.	395/200.57

#### OTHER PUBLICATIONS

Maria M. Larrondo-Petrie et al, "A Domain Analysis of Web Browser Architectures Language and Features", IEEE Southcon/96 Conference Record, pp. 168–174, Jun. 1996.

Stuart goose et al, "An Open Framework for Integrating Widely Distributed Hyper Media Resources", IEEE Multimedia Computing and System, 1996 Int'l. Conference, pp. 364–371, Jun. 1996.

Kin-Man Chung and Herbert Yuen, "A 'Tiny' Pascal Compiler: Part 1: the P-Code Interpreter," *BYTE Publications, Inc.*, Sep. 1978, pp. 59–65 and 148–155.

Kin-Man Chung and Herbert Yuen, "A 'Tiny' Pascal Compiler: Part 2: The P-Compiler," *BYTE Publications, Inc.*, Oct. 1978, pp. 34–52 (even pages only).

Ken Thompson, Regular Expression Search Algorithm, *Communications of the ACM*, vol. II, No. 6, Jun. 1968 at 419 et seq., pp. 419–422.

James G. Mitchell, William Maybury and Richard Sweet, *Mesa Language Manual*, a Xerox Corporation Document.

Gene McDaniel, *An Analysis of a Mesa Instruction Set*, a Xerox Corporation Document.

Kenneth A. Pier, *A Retrospective on the Dorado, A High-Performance Personal Computer*, a Xerox Corporation document.

Kenneth A. Pier, *A Retrospective on the Dorado, A High-Performance Personal Computer*, a Xerox Corporation document, pp. 252–269.

R.L. Johnston, Association for Computing Machinery, *The Incremental Compiler of APL/3000*, May–Jun. 1979, pp. 83–87.

Fifth Annual ACM Symposium, *Principles of Programming Languages*, Jan. 1978, pp. iii, 3–6.

Krasner, Glenn, *The Smalltalk-80 Virtual Machine*, BYTE Publications Inc., Aug. 1991, pp. 300–320.

*Primary Examiner*—Parshotam S. Lall

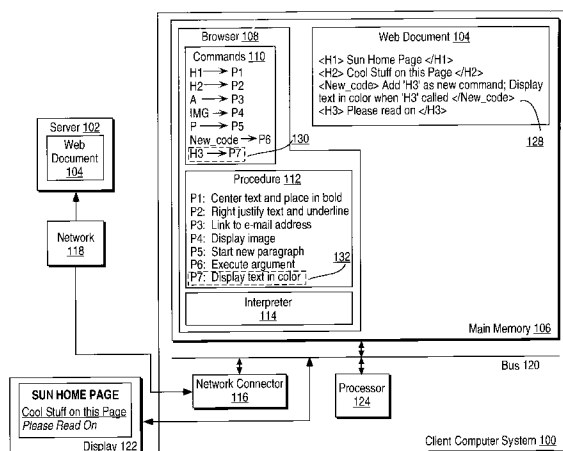
*Assistant Examiner*—Bharat Barot

*Attorney, Agent, or Firm*—Blakely Sokoloff Taylor & Zafman

#### [57] **ABSTRACT**

A method for dynamically adding new functionality to a first set of instructions that processes Web documents. The invention includes the first step of the first set of instructions decoding a first statement of the Web document. The first statement includes a first command and at least one instruction provided as an argument to the first command. In response to executing the first command, the first set of instructions decodes the instruction provided as the argument to the first command and issues the instruction to be executed. Executing the instruction provided as an argument to the first command, results in new Web document processing functionality being added to the first set of instructions.

**34 Claims, 2 Drawing Sheets**



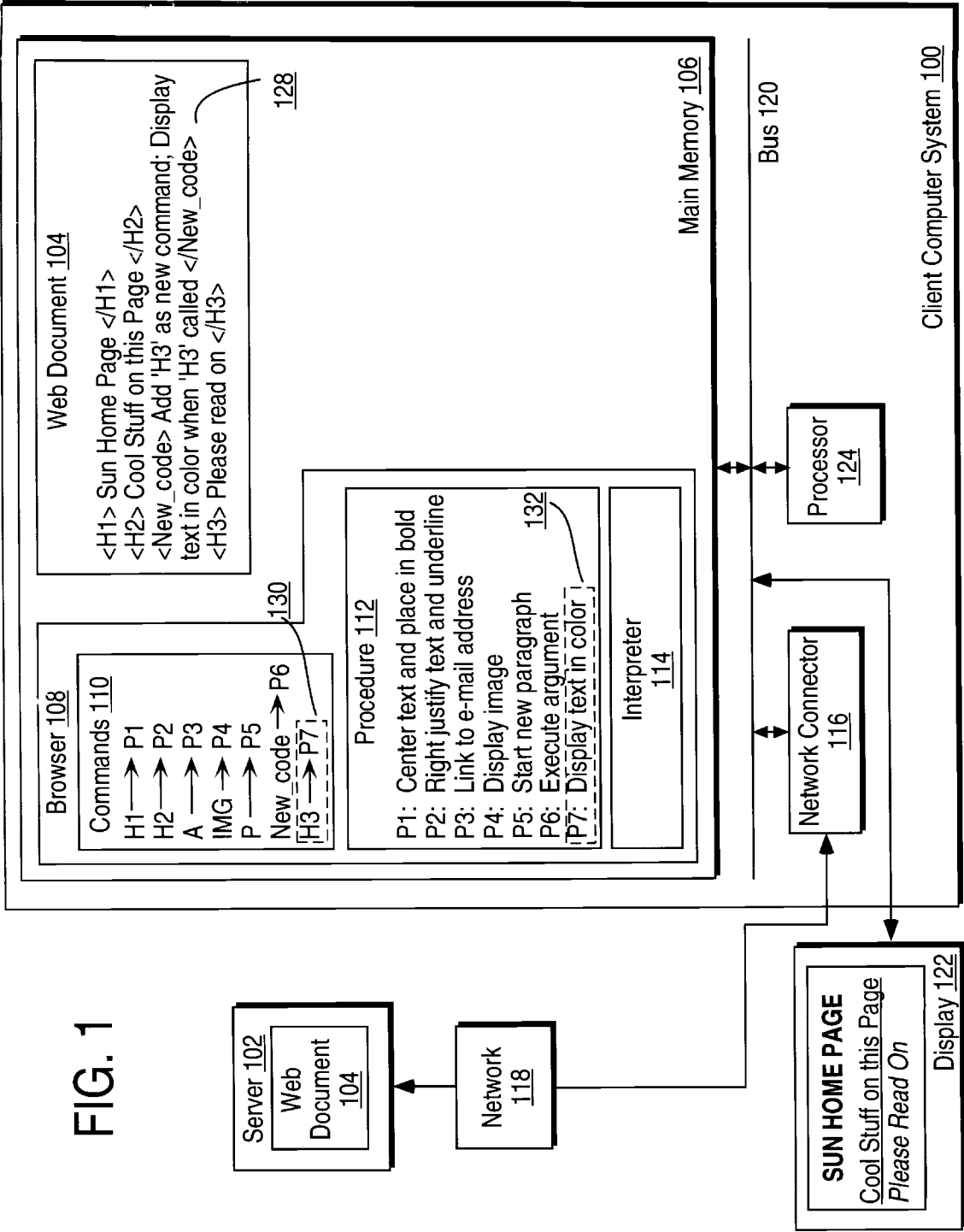
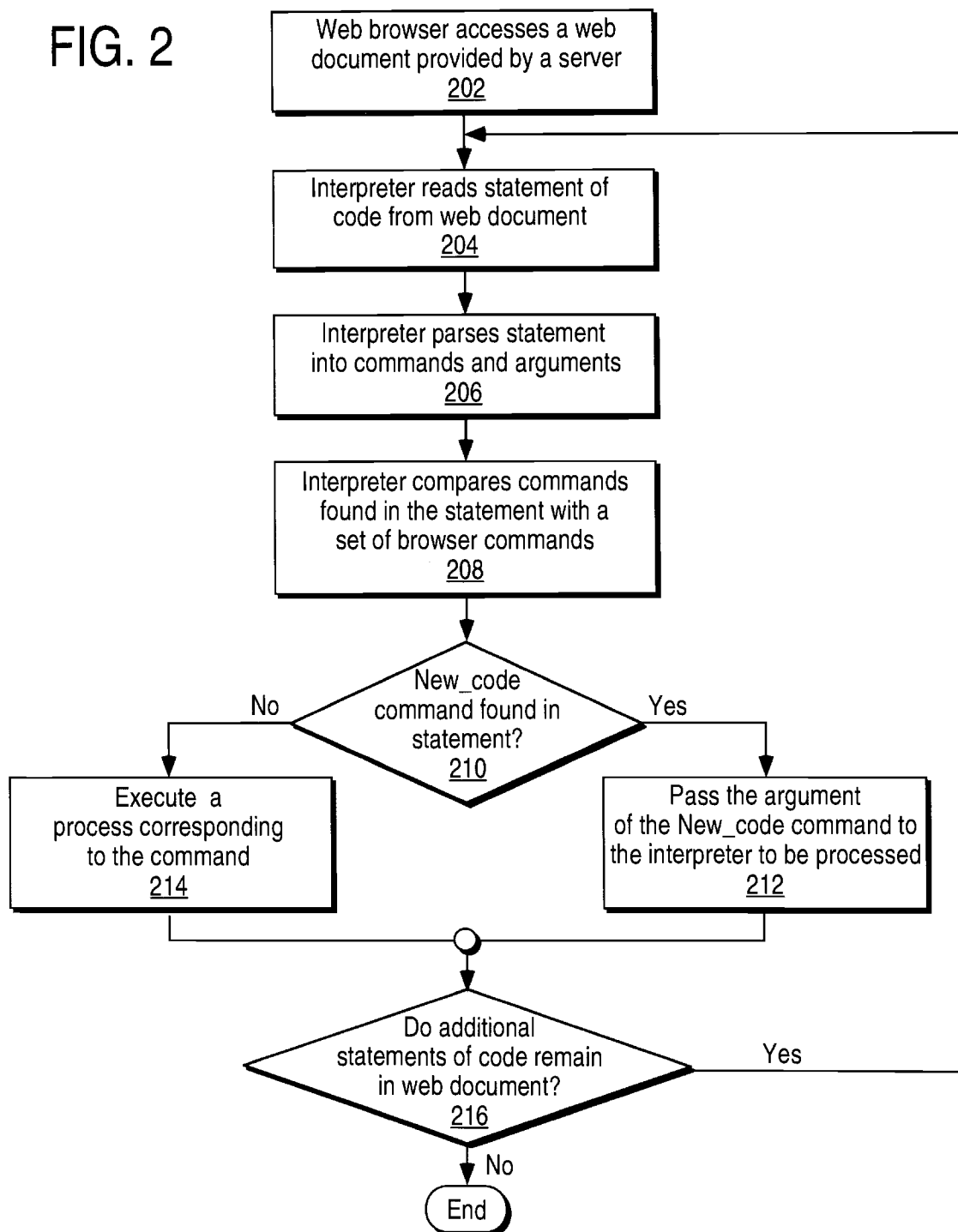


FIG. 2





**METHOD AND APPARATUS FOR  
DYNAMICALLY ADDING FUNCTIONALITY  
TO A SET OF INSTRUCTIONS FOR  
PROCESSING A WEB DOCUMENT BASED  
ON INFORMATION CONTAINED IN THE  
WEB DOCUMENT**

**FIELD OF THE INVENTION**

The present invention relates to data processing in computer systems, and in particular, dynamically adding functionality to a Web browser from a Web document.

**BACKGROUND OF THE INVENTION**

Networking technology has developed a large network of networks, referred to as the Internet, which interconnects millions of computers around the world. The Internet allows the transfer of data between any number of computer systems connected to the Internet using the Transmission Control Protocol/Internet Protocol (TCP/IP). Computers responding to service requests from other computers, via the Internet, are commonly referred to as servers, and computers that initiate requests for service from a server are referred to as clients.

The Internet has become very popular in part due to the World Wide Web (WWW), which is a network of links to hypertext documents operating within the Internet. These hypertext documents are referred to as either Web documents, Web pages, or hypertext documents. Web documents are embedded with directly accessible connections or links to other documents which create a non-linear way of reading the document. The links are embedded in Web documents as a phrase of text or an image which can be selected and activated by a computer user. Information about the Web documents are controlled and provided by Web servers. At the user's end, a Web client takes the user's requests and passes them on to the Web server.

The Web documents are written with a high level programming language referred to as the Hypertext Markup Language (HTML). Commands of the HTML, hereinafter referred to as tags, provide a variety of functions including, but not limited to, defining special format and layout information in a Web document, embedding images and sound in a Web document, and embedding links to other Web documents.

In order to access, process, and display a Web document, a client uses a first set of instructions, referred to as a browser. The browser typically includes a set of browser commands corresponding to the tags available in the HTML. Each browser command in turn points to a procedure of one or more instructions defining the command which, when executed, provide a functionality of the respective command. If the client requires service from the Web server, the browser uses the Hypertext Transfer Protocol (HTTP) to communicate with the server.

The browser compares each tag found embedded in a Web document with the set of browser commands. Once a match is found, the browser executes the procedure corresponding to the matched browser command in order to provide the functionality of the respective command.

The instructions of the browsers are typically written with a programming language different from the HTML, which includes a library of several routines. The library of routines can in turn be used to develop and add new browser commands, or modify existing browser commands, which can be embedded in a Web document as new or modified tags to provide new functionality when displaying the Web document.

The need for new browser commands, or to modify an existing browser command, is typically recognized (at the site of the server) when writing a Web document. For example, a programmer writing a Web document may want to use a new tag to display selected text on a Web document in color. However, before the new command can be embedded in a Web document as a new tag and properly executed, a browser command corresponding to the new tag must first be defined and implemented on the client's browser which is going to access and display the Web document.

Conventionally, instructions defining new browser commands, or modifying existing browser commands, are manually added to the browser by the client when the browser is not in use (i.e., offline). However, this procedure is inefficient because the client must take the time to manually add the new instructions every time a new command, or modified existing command, is to be added to the browser, regardless of whether the new or modified command will be used in other Web documents. Moreover, the procedure is impractical because the client may not have access to the instructions defining the new or modified command, and usually doesn't know when it is necessary to add instructions to define new or modify existing commands.

It is also impractical for a server to add the instructions to a client's browser because servers are generally unaware of which potential clients are going to access their Web documents. Moreover, a server rarely has access privileges to enter a client's browser and add additional instructions.

**SUMMARY OF THE INVENTION**

One embodiment of the invention provides a method for dynamically adding new functionality to a first set of instructions that processes Web documents. The invention includes the first step of the first set of instructions decoding a first statement of the Web document. The first statement includes a first command and at least one instruction provided as an argument to the first command. In response to executing the first command, the first set of instructions decodes the instruction provided as the argument to the first command and issues the instruction to be executed. Executing the instruction provided as an argument to the first command, results in new Web document processing functionality being added to the first set of instructions.

**BRIEF DESCRIPTION OF THE DRAWINGS**

One embodiment of the invention is illustrated by way of example and not limitation in the figures of the accompanying drawings, in which like references indicate similar elements, and in which:

FIG. 1 illustrates a computer system, connected to a server, wherein the computer system is capable of dynamically adding or modifying instructions in a Web browser, according to one embodiment of the invention.

FIG. 2 illustrates a flow diagram illustrating the steps of adding instructions to a Web browser from a Web document when the Web browser is interpreting the Web document, according to one embodiment of the invention.

**DETAILED DESCRIPTION**

One embodiment of the invention provides a method and apparatus for dynamically adding or modifying functionality of a Web browser. In the following description, for the purposes of explanation, numerous specific details are set forth in order to provide a thorough understanding of the

present invention. It will be apparent, however, to one skilled in the art that the present invention may be practiced without these specific details. In other instances, well-known structures and devices are shown in block diagram form in order to avoid unnecessarily obscuring the present invention.

FIG. 1 illustrates a block diagram of a computer system 100 capable of adding functionality to a Web browser 108 from a Web document 104 when the Web browser 108 is processing the Web document 104, according to one embodiment of the invention. The computer system 100 includes a Web browser 108 stored in main memory 106 for accessing and processing the Web document 104 provided by a server 102. The computer system 100 further includes a processor 124 and a network connector 116 coupled to a bus 120. The network connector 116 is further coupled to a network 118.

In one embodiment of the present invention, the browser 108 is written in the Tool Command Language (Tcl) programming language, which is an interpreted language (i.e. each statement of code is decoded and executed before the next line of code). In alternative embodiments, the browser may be written in other interpreted languages that allow extension of applications.

As illustrated in FIG. 1, computer system 100 is shown accessing, interpreting, and displaying a Web document 104 provided by the server 102. After the Web document 104 is transferred from the server 102 to the main memory 106 of client 100, the Web document 104 is interpreted by the client 100.

In the example shown in FIG. 1, the content of sample Web document 104 is illustrated within main memory 106 as including instructions to be added to the Web browser 108, which define a new browser command (i.e. the argument to the <New\_Code> 128 command) to be added to the browser's 108 set of browser commands 110. The interpreted Web document 104 is in turn shown on display 122, displaying the functionality of the new browser command. The novel way in which the present invention enables this dynamic modification of a browser is described in detail in the following sections.

FIG. 2 illustrates a flow diagram of the steps performed in order to add functionality to a Web browser from a Web document when the Web browser is processing the Web document, according to one embodiment of the present invention. In block 202, the browser 108 stored in computer system 100 accesses a Web document 104 provided by the server 102. In block 204, the interpreter 114 of the browser 108 reads a single statement of code provided in the Web document 104. In block 206, the interpreter 114 parses the statement into tags and arguments.

In block 208, the interpreter 114 compares any tags found in the statement with the set of browser commands 110 provided in the browser 108. In one embodiment of the invention, the set of browser commands 110 includes a command, hereinafter referred to as the "New\_" command, which accepts additional instructions as its argument. When executed, the New\_ command causes the instructions provided as its argument to be passed to the interpreter 114, wherein the instructions are parsed and executed. In one embodiment of the invention, the New\_ command is generated in part using the conventional "eval" command of the Tcl programming language.

In decision block 210, it is determined whether the statement of code read from the Web document includes the New\_ command. If the New\_ command is included in the

statement, then in block 212, the instructions provided as the argument to the New\_ command are passed to the interpreter 114, where they are parsed and executed as previously described.

In one embodiment of the invention, the instructions provided as the argument to the New\_ command may define a new command or modify an existing browser command. The new command, or modified existing command, may provide new functionality to the browser, which typically includes, but is not limited to, modifying an appearance of a Web document, embedding multimedia data in a Web document, and/or embedding in a Web document a link to a separate discrete unit of information.

The instructions provided as the argument to the New\_ command are encoded using the library of routines belonging to the conventional programming language used to write the Web browser 108. The new or modified browser commands may correspond to new HTML tags, which may be used to modify the appearance of a Web document, define special formats and layout information of a Web document, embed images and sound in a Web document, or embed links to other Web documents.

The new or modified existing browser commands may be enforced for the duration the present Web document is displayed, or optionally, enforced for the life of the browser 108. The instructions provided as an argument to a New\_ command may also be used to replace the client's existing browser with a new browser.

In an alternative embodiment, the instructions provided as the argument to the New\_ command may include a Uniform Resource Locator (URL), which provides an address to a separate Web document having instructions that define a new command or modify an existing browser command. The instructions may be downloaded to the client from the separate document and processed as described in block 212.

FIG. 1 illustrates an example of using the New\_ command to add a new command to the browser. As illustrated, the third statement in the Web document 104, shown stored in the main memory 106, includes a New\_ command 128 followed by the instructions to add a new browser command, "H3", which when executed will cause selected text to be displayed in color.

In response to executing the New\_ command, the interpreter adds the command H3 130 to the set of browser commands 110, and further adds the definition 132 of H3 to the set of procedures 112. The H3 command is then used as a tag in the next statement of the same Web document to have selected text of the Web document displayed in color, as indicated on display 122.

In addition, in FIG. 1, the Browser 108, which includes the Interpreter 114, the Procedures 112, the Commands 110, could alternatively be stored on another computer-readable medium, including magnetic and optical disk, which could be accessed via a disk drive coupled to the client's computer system.

In decision block 210, if it is determined that the command embedded in the statement being interpreted is not a New\_ command, then in block 214, the interpreter 114 executes the procedure corresponding to the command in order to provide the functionality of the command.

In decision block 216, it is determined whether additional statements remain to be processed in the Web document 104. If no statements remain to be processed, the Web document remains displayed until a user of the computer system 100 exits the Web document using conventional means. On the other hand, if statements remain to be processed, the remain-

5,845,075

5

ing statements are sequentially processed in steps similar to those described in blocks 204–216, wherein the individual statements are parsed and executed in order to provide the functionality of each command embedded in the statement.

The apparatus and method of the present invention for dynamically adding or modifying functionality of a browser from a Web document when the browser is interpreting the Web document provides many advantages. For example, servers can provide Web documents using new tags and the browser commands corresponding to the new tags without regard to the commands available on a client's browser and be assured that the tags will be executed as defined by the server.

The present invention also allows the browser commands corresponding to the new tags to be transparently added to the client's browser. As a result, the client does not have to know when, or how, a new or modified command is to be defined on their browser.

In addition, the present invention can also be used to conserve memory space, which may be critical on a limited resource set top box. For example, a new or modified existing browser command may only be applicable to a new tag being used in the present Web document. In such a case, the functionality to be added to a Web browser can be defined to only remain on the browser for the duration of the present Web document's display.

In the foregoing specification the invention has been described with reference to specific exemplary embodiments thereof. It will, however, be evident that various modifications and changes may be made thereto without departing from the broader spirit and scope of the invention. The specification and drawings are, accordingly, to be regarded in an illustrative rather than restrictive sense. Moreover, the following claims indicate the scope of the invention, and all variations which come within the meaning and range of equivalency of the claims are to be embraced within their scope.

What is claimed is:

1. A computer implemented method for adding functionality to a first set of instructions provided for processing a Web document, said method comprising the steps of:

executing said first set of instructions to process said Web document;

said first set of instructions decoding a first statement provided by said Web document, said first statement including a first command and at least one instruction provided as an argument to said first command;

said first set of instructions issuing said first command provided by said Web document to be executed;

in response to an execution of said first command, said first set of instructions decoding said one instruction provided as said argument to said command and issuing said one instruction to be executed; and

providing new Web document processing functionality to said first set of instructions upon execution of said one instruction provided by said first statement of said Web document.

2. The computer-implemented method of claim 1, wherein the first set of instructions is a Web browser and said statement of said Web document is generated in a Hypertext Markup Language (HTML).

3. The computer-implemented method of claim 1, further including the step of:

prior to said step of decoding said first statement, retrieving, via a network connection, said Web document from a separate computer system.

6

4. The computer-implemented method of claim 3, wherein said first set of instructions includes a set of commands, wherein each command provides a separate functionality selected from a group of functionalities consisting of modifying an appearance of said Web document, embedding multimedia data in said Web document, and embedding in said Web document a link to a separate discrete unit of information.

5. The computer-implemented method of claim 4, wherein said set of commands are HTML commands.

6. The computer-implemented method of claim 4, wherein in further response to said execution of said one instruction provided as said argument to said first command, adding a new command to said set of commands, said new command having a functionality selected from said group of functionalities.

7. The computer-implemented method of claim 6, further comprising the steps of:

said first set of instructions decoding a second statement of said Web document, said second statement including said new command;

said first set of instructions issuing said new command to be executed;

in response to an execution of said new command, invoking said functionality of said new command during a processing of said Web document.

8. The computer-implemented method of claim 4, wherein in further response to said execution of said one instruction provided as said argument to said first command, modifying a functionality of an existing command in said set of commands.

9. The computer-implemented method of claim 8, further comprising the steps of:

said first set of instructions decoding a third statement of said hypertext document, said third statement including said existing command;

said first set of instructions issuing said existing command to be executed;

in response to an execution of said existing command, invoking said functionality of said existing command, as modified, during a processing of said hypertext document.

10. A computer-implemented method for enabling a target computer system to add functionality to a first set of instructions during an execution of said first set of instructions, said first set of instructions capable of processing a hypertext document having a collection of discrete units of information, wherein each discrete unit is capable of including a link to a separate discrete unit of information, by transmitting a sequence of instructions from a host computer system which, when executed, cause said target computer system to perform the method recited in any one of the claims 7 or 9.

11. A computer-readable medium having stored thereon a first set of instructions, said first set of instructions provided for processing a Web document, said first set of instructions, which, when executed by a processor, cause said processor to perform the steps of:

decoding by said first set of instructions a first statement provided by said Web document, said first statement including a first command and at least one instruction provided as an argument to said first command;

issuing by said first set of instructions said first command provided by said Web document to be executed;

in response to an execution of said first command, decoding by said first set of instructions said one instruction



provided as said argument to said command and issuing said one instruction to be executed; and

providing new Web document processing functionality to said first set of instructions upon execution of said one instruction provided by said first statement of said Web document. 5

12. The computer-readable medium of claim 11, wherein the first set of instructions is a Web browser and said statement of said Web document is generated in a Hypertext Markup Language (HTML). 10

13. The computer readable medium of claim 11, containing additional instructions in said first set of instructions which, when executed by said processor, cause said processor to perform the steps of:

prior to said step of decoding said first statement, retrieving, via a network connection, said Web document from a separate computer system. 15

14. The computer-readable medium of claim 13, wherein said first set of instructions includes a set of commands, wherein each command provides a separate functionality selected from a group of functionalities consisting of modifying an appearance of said Web document, embedding multimedia data in said Web document, and embedding in said Web document a link to a separate discrete unit of information. 20

15. The computer-readable medium of claim 14, wherein said set of commands are HTML commands.

16. The computer-readable medium of claim 14, wherein in further response to said execution of said one instruction provided as said argument to said first command, adding a new command to said set of commands, said new command having a functionality selected from said group of functionalities. 25

17. The computer-readable medium of claim 16, containing additional instructions in said first set of instructions which, when executed by a processor, cause said processor to perform the steps of:

decoding by said first set of instructions a second statement of said Web document, said second statement including said new command; 30

issuing by said first set of instructions said new command to be executed;

in response to an execution of said new command, invoking said functionality of said new command during a processing of said Web document. 35

18. The computer-readable medium of claim 14, wherein in further response to said execution of said one instruction provided as said argument to said first command, modifying a functionality of an existing command in said set of commands. 40

19. The computer-readable medium of claim 18, containing additional instructions in said first set of instructions which, when executed by a processor, cause said processor to perform the steps of:

said first set of instructions decoding a third statement of said Web document, said third statement including said existing command; 45

said first set of instructions issuing said existing command to be executed; 50

in response to an execution of said existing command, invoking said functionality of said existing command, as modified, during a processing of said Web document.

20. A computer system comprising of:

a first mechanism to process a Web document, said first mechanism operable to decode a first statement provided by said Web document, said first statement

including a first command and at least one instruction provided as an argument to said first command;

said first mechanism operable to issue said first command provided by said Web document to a processor to be executed; and

said first mechanism operable to decode, in response to an execution of said first command, said one instruction provided as said argument to said command and issue said one instruction to be executed, wherein execution of said one instruction provides new Web document processing functionality to said first mechanism.

21. The computer system of claim 20, wherein the first statement of the Web document is generated in a Hypertext Markup Language (HTML).

22. The computer system of claim 20, further comprising: a second mechanism configured to retrieve, via a network connection, said Web document from a separate computer system.

23. The computer system of claim 22, wherein said first mechanism includes a set of commands, wherein each command provides a separate functionality selected from a group of functionalities consisting of modifying an appearance of said Web document, embedding multimedia data in said Web document, and embedding in said Web document a link to a separate discrete unit of information. 25

24. The computer system of claim 23, wherein said set of commands are HTML commands.

25. The computer system of claim 23, wherein said first mechanism is further configured to add, in response to said execution of said one instruction provided as said argument to said first command, a new command to said set of commands, said new command having a functionality selected from said group of functionalities.

26. The computer system of claim 25, wherein said first mechanism is further configured to decode a second statement of said Web document, said second statement including said new command, said first mechanism configured to issue said new command to be executed, and said first mechanism is further configured to invoke, in response to an execution of said new command, said functionality of said new command during a processing of said Web document. 30

27. The computer system of claim 23, said first mechanism further configured to modify, in response to an execution of said one instruction provided as said argument to said first command, a functionality of an existing command in said set of commands. 35

28. The computer system of claim 27, wherein said first mechanism is further configured to decode a third statement of said Web document, said third statement including said existing command, said first mechanism configured to issue said existing command to be executed, and said first mechanism further configured to invoke, in response to an execution of said existing command, said functionality of said existing command, as modified, during a processing of said Web document. 40

29. A machine-implemented method comprising the steps of:

receiving a Web document, the Web document providing information indicating a new functionality for processing the Web document; 45

executing a set of instructions to process the document, the set of instructions including a set of one or more commands that provide functionality for processing the Web document; 50

the set of instructions processing the information provided by the Web document, which information indicates the new functionality for processing the Web document; and

5,845,075

9

in response to processing the information provided by the Web document for indicating the new functionality for processing the Web document, altering the set of one or more commands of the set of instructions to include a new command for providing the new functionality to the set of instructions for processing the Web document. 5

30. The method of claim 29, wherein the set of instructions comprises a Web browser.

31. The method of claim 30, wherein at least a portion of the Web document is generated in a Hypertext Markup Language (HTML). 10

32. The method of claim 29, further comprising the steps of:

decoding a first statement contained in the information, 15

the first statement including the new command and at

10

least one instruction provided as an argument to the new command;

the set of instructions issuing the new command to be executed; and

in response to execution of the new command, the set of instructions decoding the at least one instruction provided as an argument to the new command and issuing the one instruction to be executed to add the new functionality to the set of instructions for processing the Web document.

33. The method of claim 32, wherein the the set of instructions comprises a Web browser.

34. The method of claim 29, wherein the new command represents a modified one or more of the set of commands.

\* \* \* \* \*

United States Patent [19]  
Nielsen

[11] Patent Number: 5,903,727  
[45] Date of Patent: May 11, 1999

[54] **PROCESSING HTML TO EMBED SOUND IN A WEB PAGE**

[75] Inventor: **Jakob Nielsen**, Atherton, Calif.

[73] Assignee: **Sun Microsystems, Inc.**, Palo Alto, Calif.

[21] Appl. No.: **08/665,487**

[22] Filed: **Jun. 18, 1996**

[51] **Int. Cl.<sup>6</sup>** ..... **G06F 13/00**

[52] **U.S. Cl.** ..... **395/200.42; 395/200.33; 395/200.47; 395/200.48**

[58] **Field of Search** ..... **395/200.41–200.43, 395/200.46–200.5, 356–357, 200, 200.33**

[56] **References Cited**

**U.S. PATENT DOCUMENTS**

5,293,385	3/1994	Hary	395/183.14
5,572,643	11/1996	Judson	395/200.48
5,606,359	2/1997	Youden et al.	348/7
5,659,729	8/1997	Nielsen	707/3
5,715,404	2/1998	Katseff et al.	395/200.62
5,737,533	4/1998	De Honda	395/200.49
5,802,292	9/1998	Mogul	395/200.49

**OTHER PUBLICATIONS**

John W. Mezer, IEEE, Oct. 1991, pp. 21–22 “Hypertext a New Dimension in Data Integration.”.

Ronald J. Vetter et al, IEEE, Oct. 1994, pp. 49–57 “Mosaic and the World Wide Web.”.

Venkata N. Padmanabhan, “Improving World Wild Web Latency”, Computer Science Devision—University of California at Berkeley, Report No. UCB/CSD–95–875, pp. 1–24, May 1995.

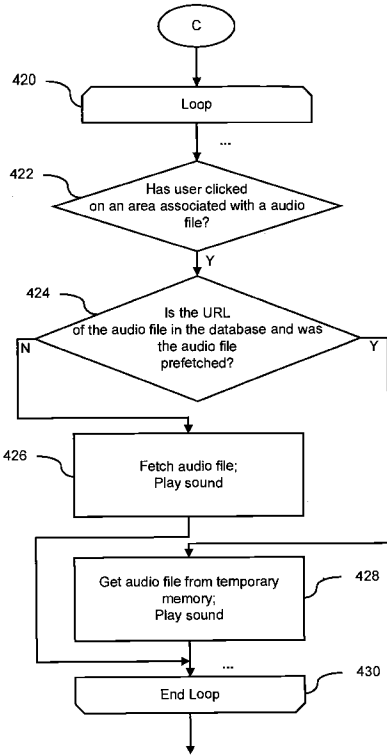
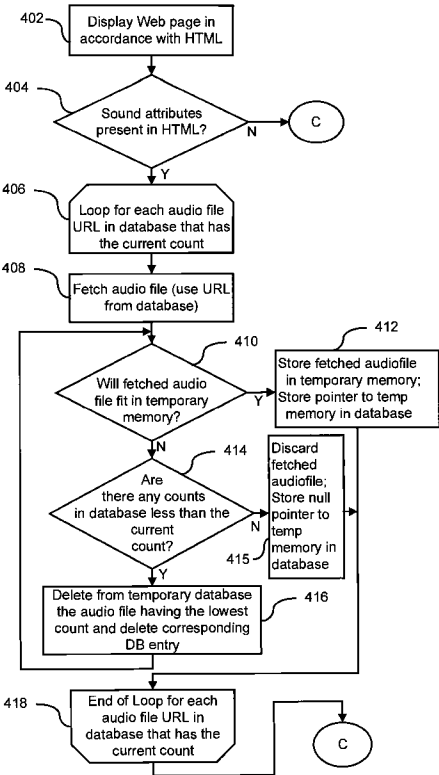
Microsoft Internet Explorer 2.0 for Windows 95, 1995. “http://www.microsoft.com/ic/win95/default.htm.”.

Primary Examiner—Parshotam S. Lall  
Assistant Examiner—Bharat Barot  
Attorney, Agent, or Firm—Graham & James LLP

[57] **ABSTRACT**

A method and apparatus that allows a Web page designer to specify that an audio file linked to a Web page should be prefetched before user input is accepted. Web browser software prefetches the audio file if there is enough room in a temporary memory to store the file. The invention also allows a Web page designer to specify the text over which the user must place the cursor to play the audio file. When the temporary memory is full and an audio file needs to be prefetched, the browser deletes files from the temporary memory until there is enough room in the temporary memory for the prefetched audio file. Files are deleted in a least-recently-referenced, first-out order.

17 Claims, 8 Drawing Sheets



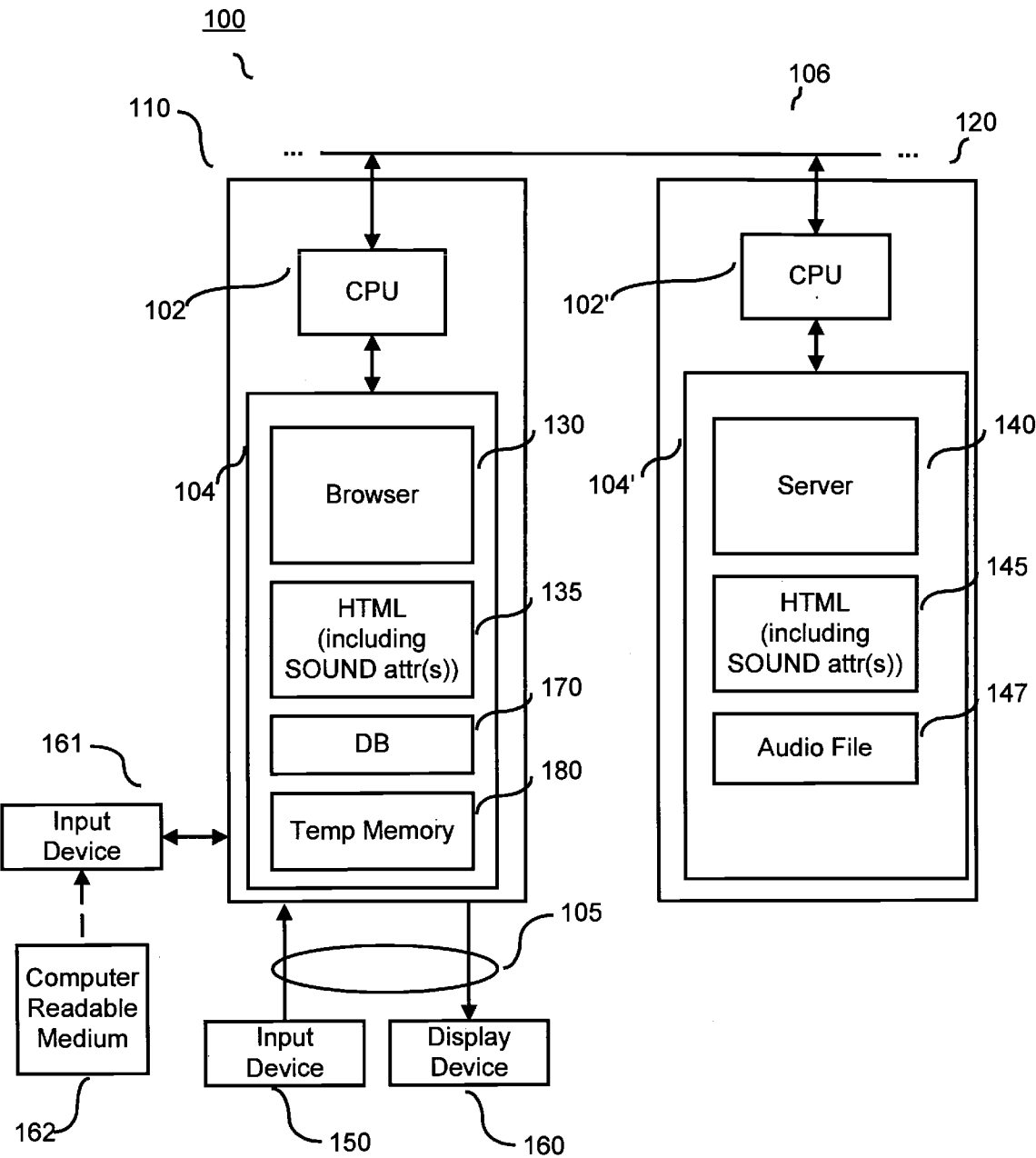


Fig. 1

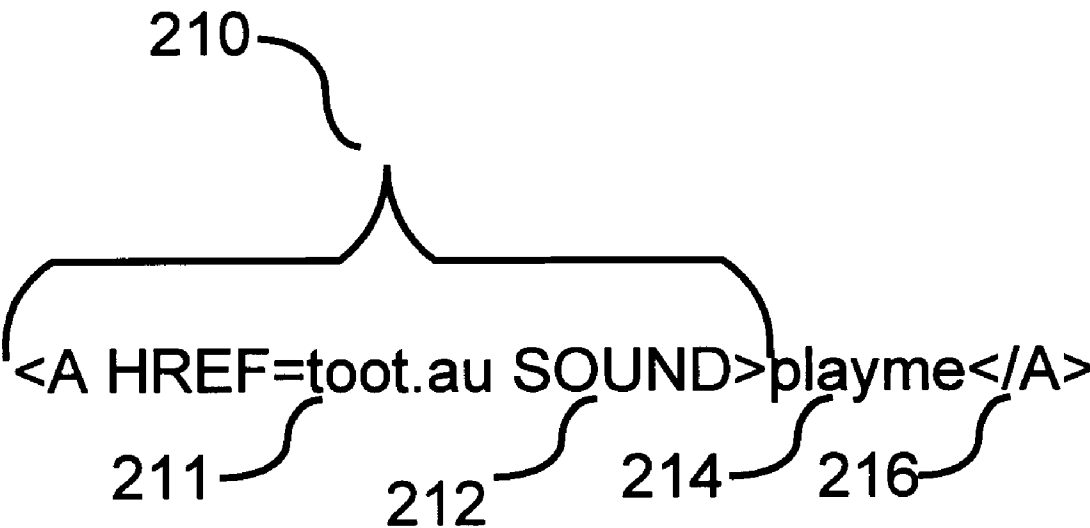


Fig. 2



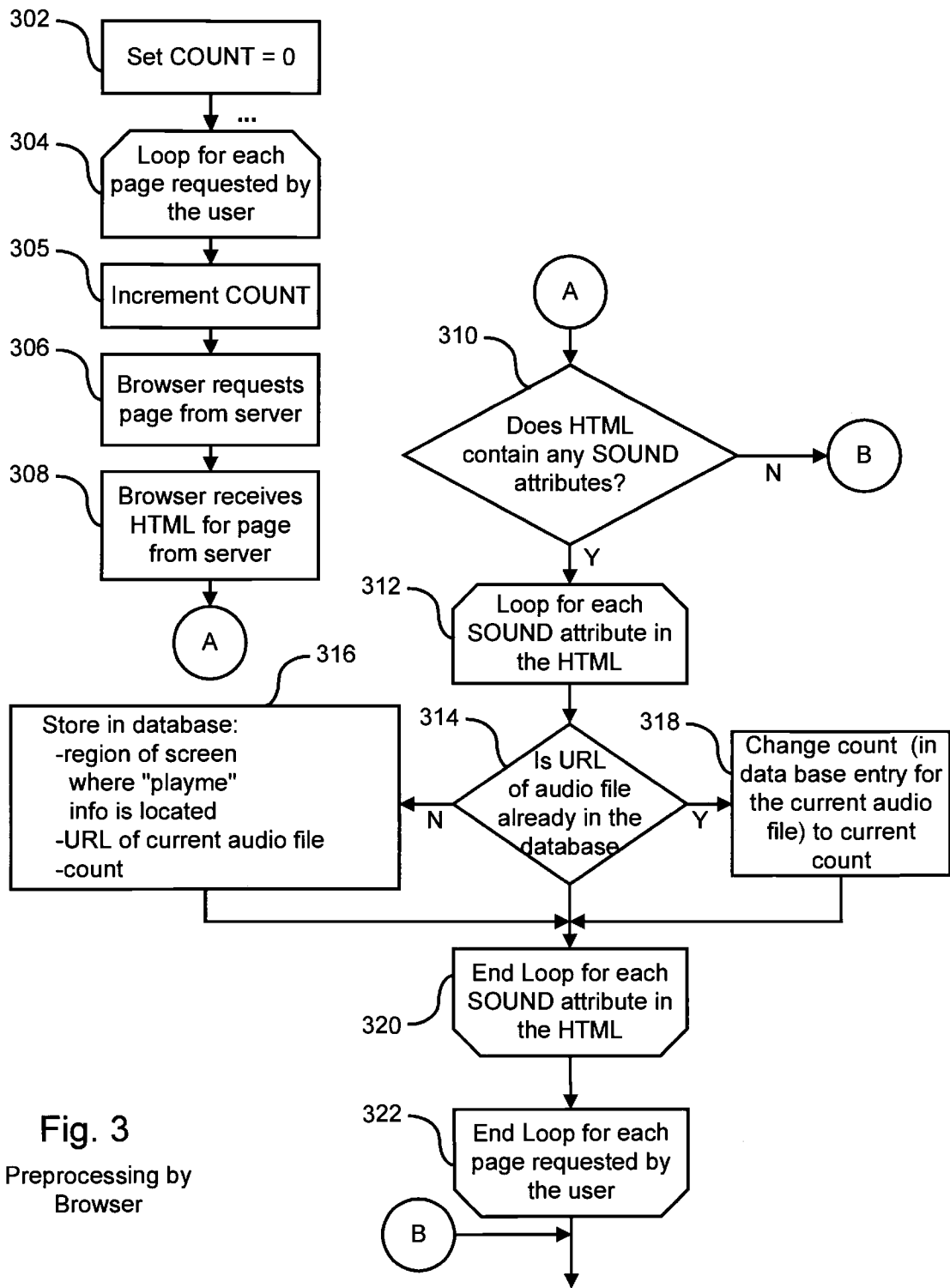
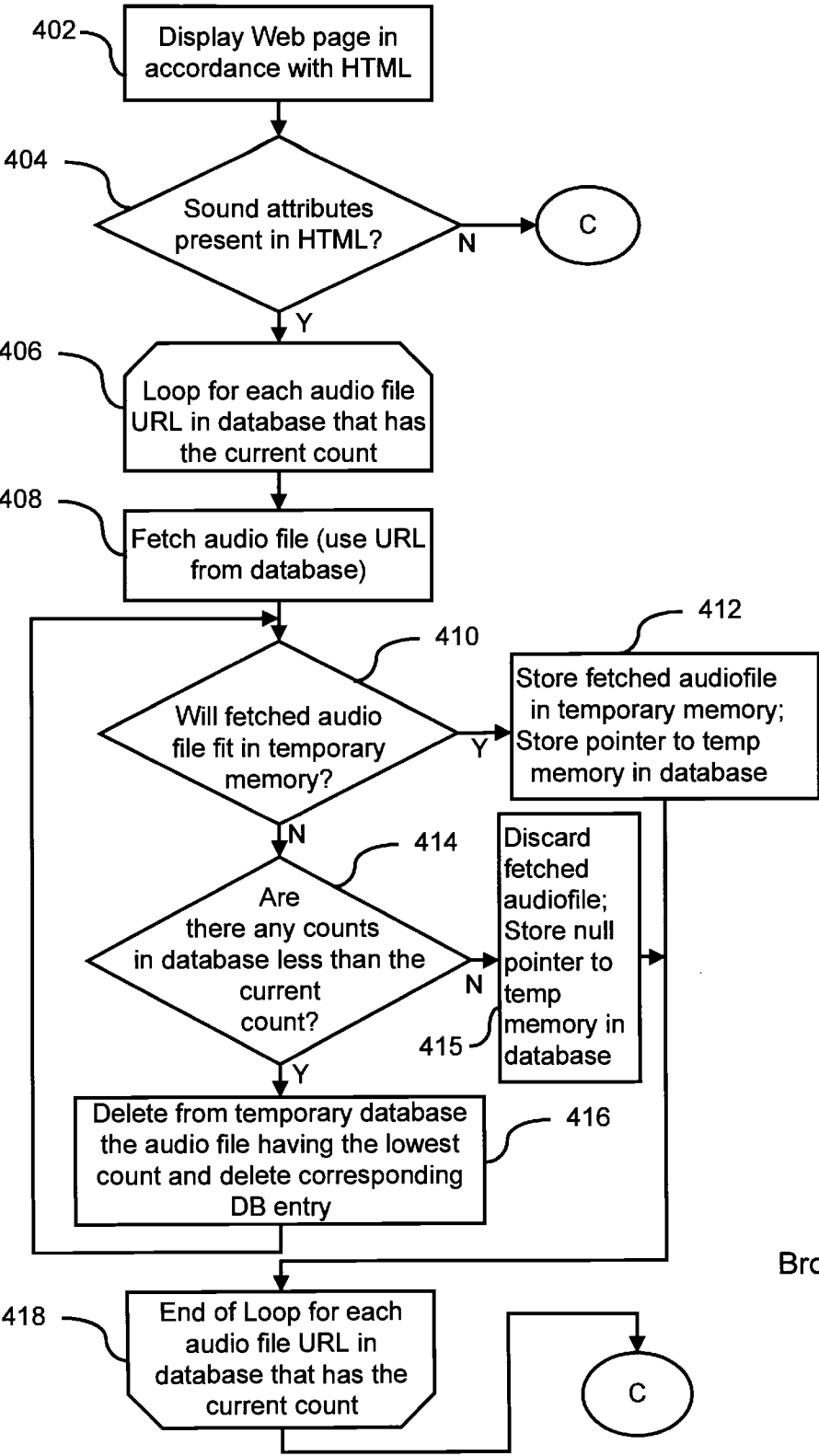


Fig. 3  
Preprocessing by  
Browser



Browser Displays  
a Page  
Fig. 4(a)

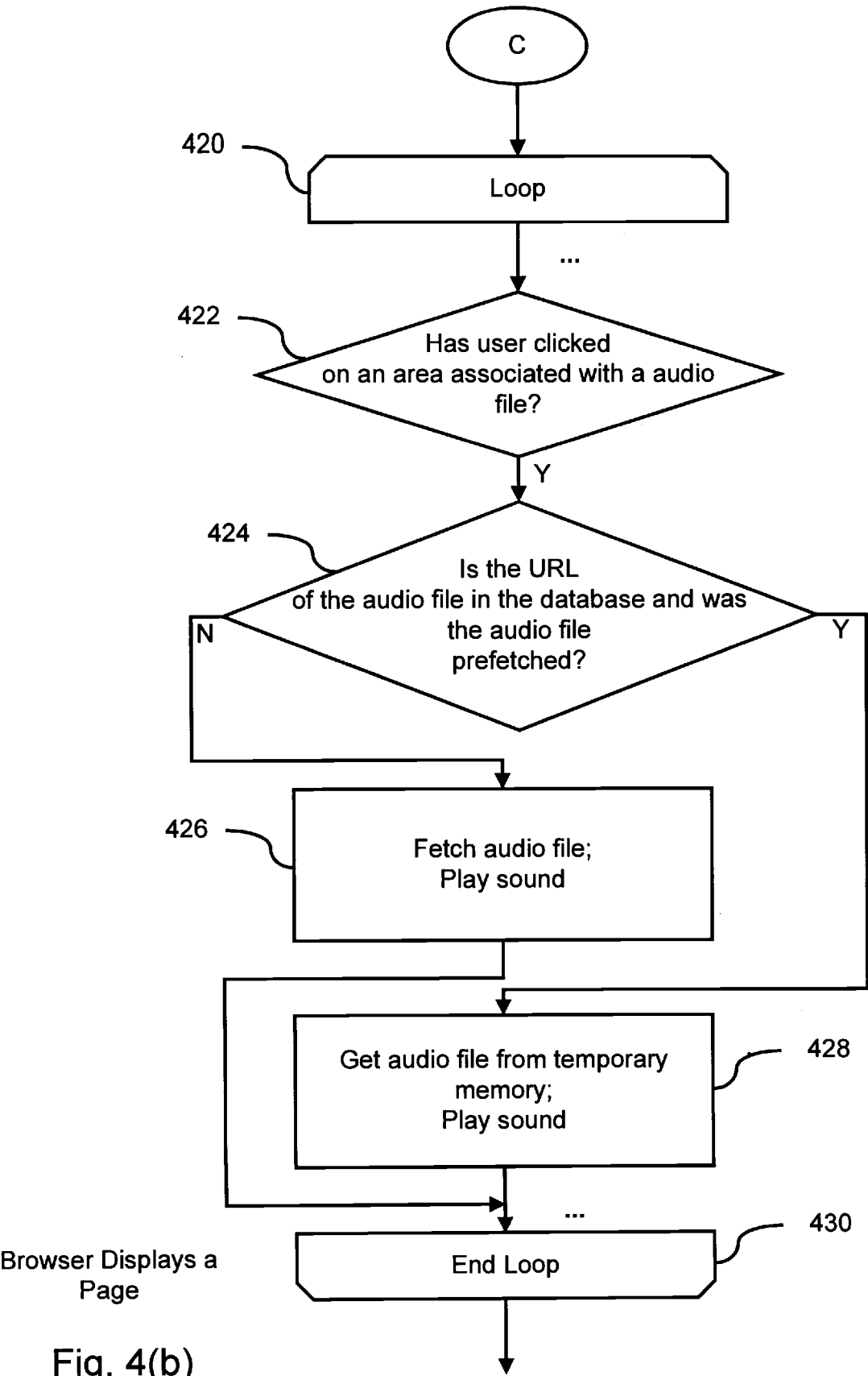
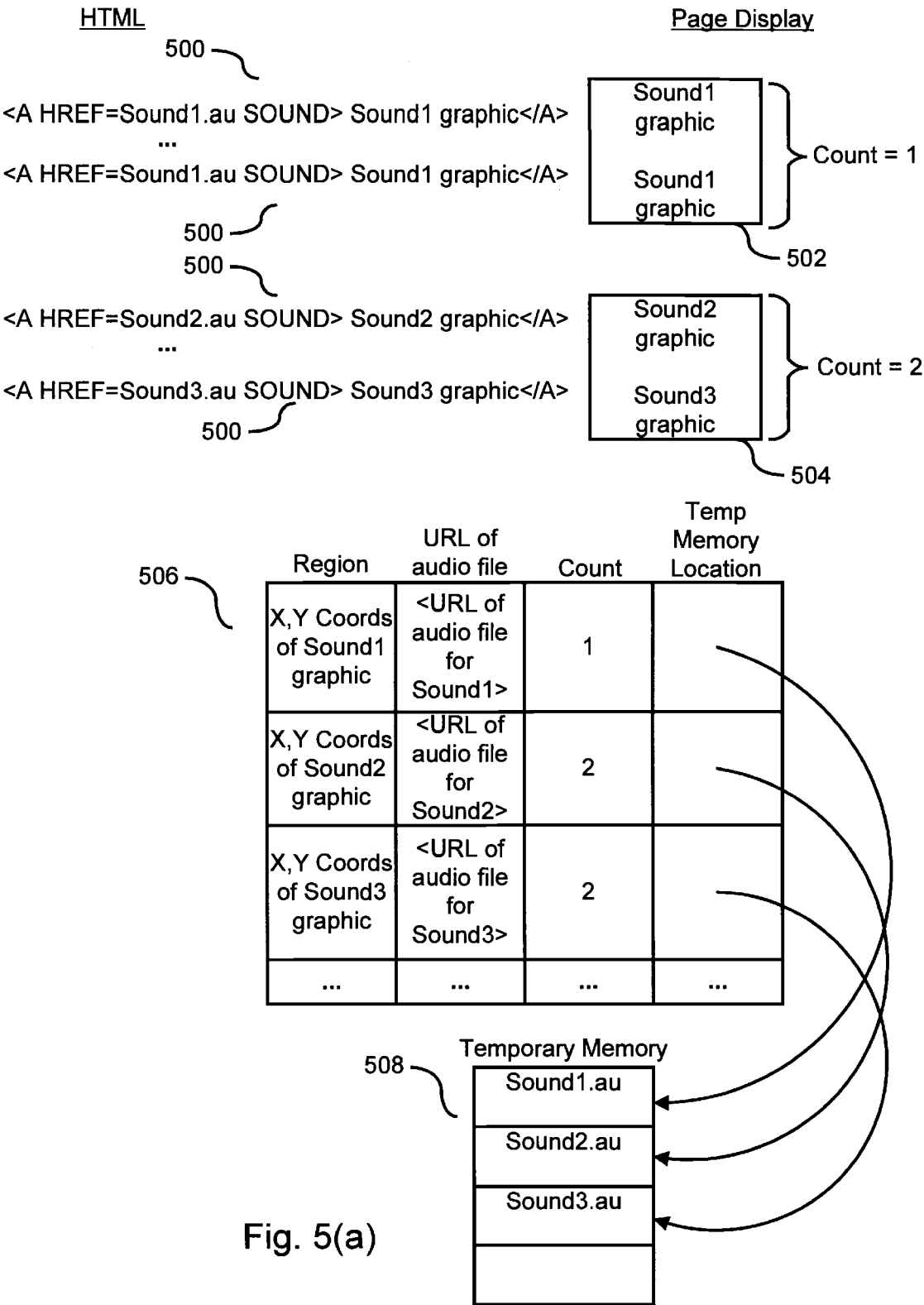


Fig. 4(b)



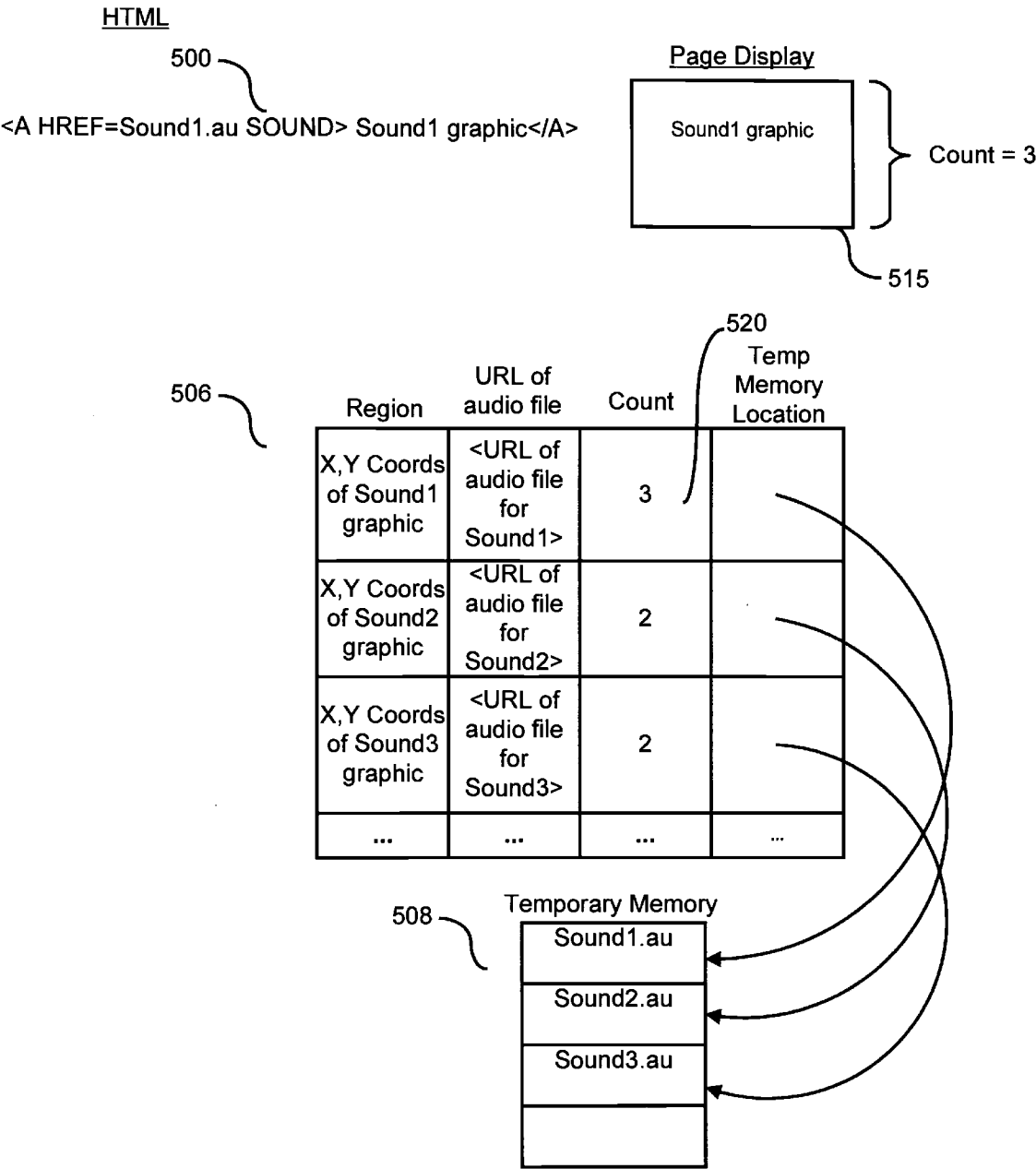


Fig. 5(b)

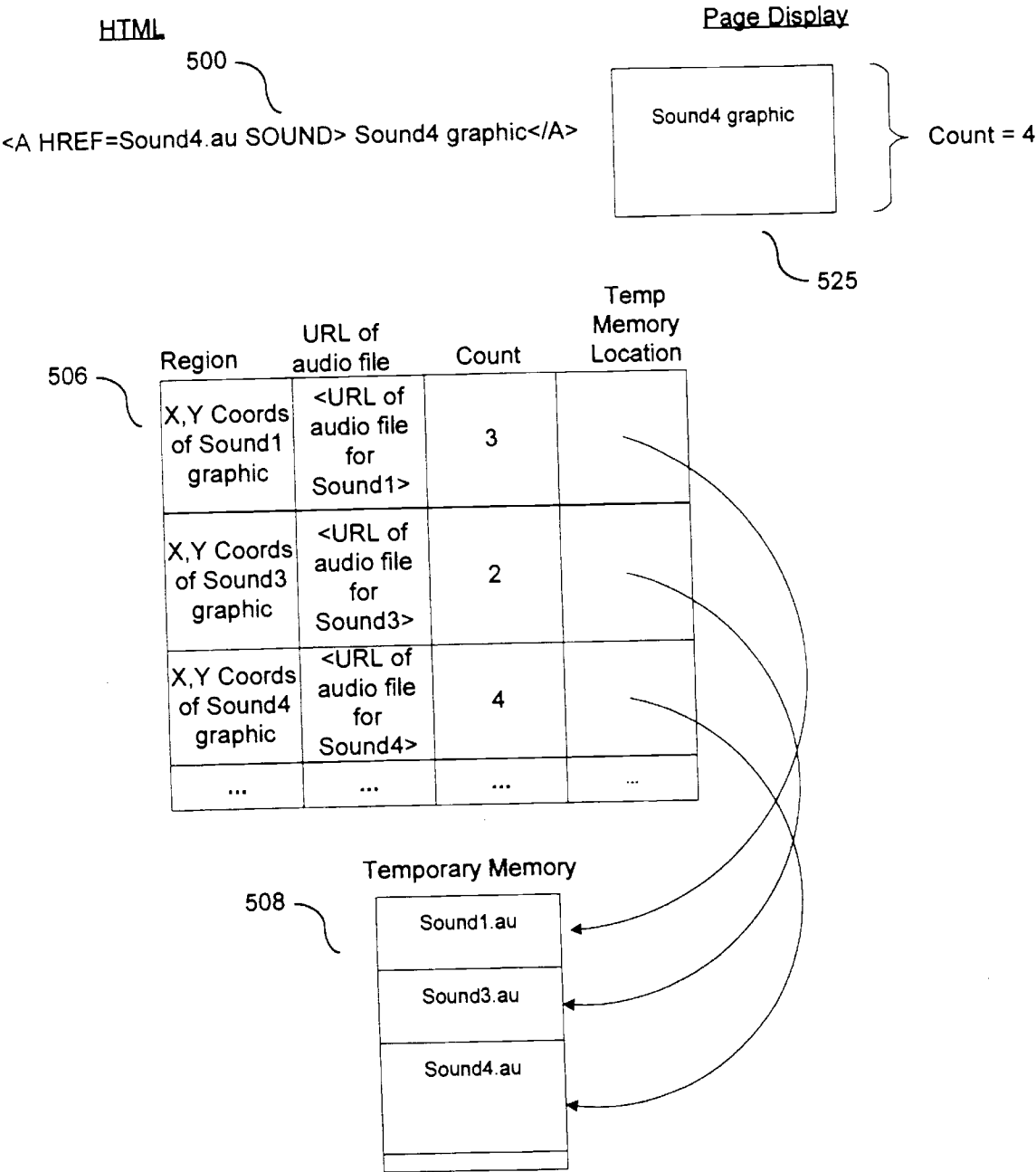


Fig. 5(c)

5,903,727

1

**PROCESSING HTML TO EMBED SOUND IN  
A WEB PAGE**

**FIELD OF THE INVENTION**

This application relates to the World Wide Web and, in particular, to a software tool for reducing the time required to play sounds in conjunction with documents on the Web.

**BACKGROUND OF THE INVENTION**

The past several years have seen an explosive growth of the internet, and specifically, in the growth of the World Wide Web (hereafter "the Web"). The Web is built around a network of "server" computers which exchange requests and data from each other using the hypertext transfer protocol ("http"). A human designer designs the layout of a Web page and specifies the layout of the page using HTML ("Hypertext Markup Language"). Several versions of HTML are currently in existence. Examples include HTML versions 2.0 and 3.0, as specified by the WWW Consortium of MIT.

A user views a Web page using one of a number of commercially available "browser" programs. The browser submits an appropriate http request to establish a communications link with a Web server of the network. A typical http request references a Web page by its unique Uniform Resource Locator ("URL"). A URL identifies the Web server hosting that Web page, so that an http request for access to the Web page can be routed to the appropriate Web server for handling. Web pages can also be linked graphically to each other.

Ever since the Mosaic browser, which was developed by the University of Illinois, it has been possible to have a sound effect as the destination for a hypertext link on the Web. When the user clicks on the area associated with the sound, the browser retrieves an audio file from the remote server and plays it. Early browsers actually played audio files by passing them to so-called "helper applications," but current browsers have audio players built in. Unfortunately, this standard approach (retrieving the audio file at a user's request) is ill-suited for small sound effects (e.g., a bird call or a name pronunciation) because of network latency. The user does not get to hear the sound until several seconds after he or she has clicked on the link. Experimentation has shown that a user needs to get the result of traversing a hypertext link within a single second for the user to feel that he or she is navigating freely.

A second approach supports so-called "background sounds" that will play automatically as soon as the browser has finished downloading a page. The problem with this approach is that the sound will be played under all circumstances, and thus not under user control when the user desires to hear a specific sound. Also, no more than a single sound can be associated with the page.

**SUMMARY OF THE INVENTION**

The present invention overcomes the problems and disadvantages of the prior art by adding an extension to HTML that specifies that certain audio files should be prefetched before user input is accepted for a displayed Web page. The invention also enables the designer to specify that only certain audio files should be prefetched. For example, it is not always desirable to prefetch a very large audio file because it would take too much time. The HTML format allows a Web page designer to specify the information over which the user must place the cursor to activate the

2

prefetched audio output. The invention also includes browser software that can prefetch audio files in accordance with instructions in the HTML.

In accordance with the purpose of the invention, as embodied and broadly described herein the invention is a method of processing HTML that describes a Web page, comprising the steps performed by a data processing system, of receiving HTML describing the Web page; reviewing the HTML to locate a SOUND attribute in the HTML, the SOUND attribute being associated with an audio file; prefetching the audio file; and storing the prefetched audio file in a memory of the data processing system.

In further accordance with the purpose of this invention, as embodied and broadly described herein the invention is a computer system that processes HTML describing a Web page, the computer system having a memory, comprising: an HTML receiving portion that receives the HTML describing the Web page; a reviewing portion that reviews the HTML to locate a SOUND attribute in the HTML, the SOUND attribute being associated with an audio file; a prefetch portion that prefetches the audio file; and a storage portion that stores the prefetched audio file in the memory of the computer system.

Objects and advantages of the invention will be set forth in part in the description which follows and in part will be obvious from the description or may be learned by practice of the invention. The objects and advantages of the invention will be realized and attained by means of the elements and combinations particularly pointed out in the appended claims.

**BRIEF DESCRIPTION OF THE DRAWINGS**

The accompanying drawings, which are incorporated in and constitute a part of this specification, illustrate several embodiments of the invention and, together with the description, serve to explain the principles of the invention.

FIG. 1 is a block diagram of a computer system in accordance with a preferred embodiment of the present invention.

FIG. 2 shows a format of a HyperText Markup Language (HTML) element in accordance with the embodiment of FIG. 1.

FIG. 3 is a flow chart showing steps performed by browser software prior to displaying a Web page.

FIG. 4 is a flow chart showing steps performed by the browser software to display a Web page.

FIG. 5(a) shows an example of contents of a temporary memory of FIG. 1.

FIG. 5(b) shows an example of contents of a temporary memory of FIG. 1.

FIG. 5(c) shows an example of a data structure stored in a memory of FIG. 1.

**DETAILED DESCRIPTION OF THE  
PREFERRED EMBODIMENT**

Reference will now be made in detail to a preferred embodiment of the invention, an example of which is illustrated in the accompanying drawings. Wherever possible, the same reference numbers will be used throughout the drawings to refer to the same or like parts.

FIG. 1 is a block diagram of a computer system 100 in accordance with a preferred embodiment of the present invention. Computer system 100 includes a first computer 110 and a second computer 120. First computer 110 and



second computer 120 are connected together via line 106, which can be, for example, a LAN, a WAN, or an internet connection. Line 106 can also represent a wireless connection, such as a cellular network connection.

First computer 110 includes a CPU 102; a memory 104; input/output lines 105; an input device 150, such as a keyboard or mouse; and a display device 160, such as a display terminal. First computer 110 also includes an input device 161 that reads computer instructions stored on computer readable medium 162. These instructions are the instructions of e.g., browser software 130. Memory 104 of first computer 110 includes browser software 130, Hypertext Markup Language (HTML) 135, and a database or data structure (DB) 170. A portion of memory 104 is designated as temporary memory 180. A person of ordinary skill in the art will understand that memory 104 also contains additional information, such as application programs, operating systems, data, etc., which are not shown in FIG. 1 for the sake of clarity.

Second computer 120 includes a CPU 102' and a memory 104'. Memory 104' of second computer 120 includes server software 140, Hypertext Markup Language (HTML) 145, and an audio file 147. HTML 135 in the memory of first computer 110 was downloaded over line 106 from HTML 145 of second computer 120. A person of ordinary skill in the art will understand that memory 104' also contains additional information, such as application programs, operating systems, data, etc., which are not shown in FIG. 1 for the sake of clarity. Server 140, HTML 145, and audio file 147 can also be located in memory 104 of first computer 110.

It will be understood by a person of ordinary skill in the art that computer system 100 can also include numerous elements not shown in the FIG. 1 for the sake of clarity, such as disk drives, keyboards, display devices, network connections, additional memory, additional CPUs, LANs, input/output lines, etc. A preferred embodiment of the invention runs under the Solaris operating system, Version 2.5. Solaris is a registered trademark of Sun Microsystems, Inc.

The present invention speeds up the process of playing sound in conjunction with Web pages. In the present invention, some audio files are specified in the HTML for a Web page with a attribute indicating that they should be prefetched if possible. When a Web page is first displayed, the browser prefetches as many of these audio files as will fit in temporary memory. Thus, when the user activates an audio file (usually by clicking on text or graphics linked to the audio file), the prefetched audio file can be played immediately without waiting for the time normally required to download the audio file over line 106. In a preferred embodiment, Web pages specified using the present invention can still be displayed using Web browsers that have not been enhanced to recognize HTML SOUND attributes because these browsers will simply ignore the SOUND attribute.

FIG. 2 shows an example of a format of an anchor element in accordance with a preferred embodiment of the present invention. The format of conventional HTML is described in, for example, Morris, "HTML for Fun and Profit," Sun-Soft Press 1995, which is herein incorporated by reference. The format includes a first text string 210 surrounded by brackets ("<" and ">"). The characters "A HREF=" are followed by a URL 211 of an audio file (e.g., toot.au). The audio file can be of any appropriate format, such as .AU or .WAV formats. The attribute "SOUND" 212 indicates that the audio file should be prefetched.

Next, a second text string 214 (e.g., playme) identifies information that is displayed on the Web page and that

should be linked to the audio file. The string 214 can identify any information that has a visual appearance on display device 160, including images, literal character strings animations (given a system of sufficient response time) and JAVA applets. JAVA is a registered trademark of Sun Microsystems, Inc. Thus, for example, audio output can be associated with an icon if the displayed information identifies a file containing the graphical information of an icon. Next, a third text string 216 consists of the string "</A>" to indicate the end of the SOUND element.

FIG. 3 is a flow chart showing steps performed by browser software 130 prior to displaying a Web page. It will be understood by persons of ordinary skill in the art that the steps of flow charts in this document are performed by CPU 102 of FIG. 1 executing the instructions of browser 130 in accordance with HTML 135. Initially, in step 302, a COUNT variable in memory 104 is set to "0." COUNT is incremented each time a new page is viewed via the browser and, therefore, keeps track of a current page number that is currently being viewed. A page having a lowest associated COUNT value was viewed the longest ago in time. Step 304 represents the top of a loop performed for each Web page viewed. In step 305, the COUNT variable is incremented so that its value represents the new page to be viewed. In step 306, the browser requests HTML 145 for a current page from server 140. In step 308, server 140 sends the requested HTML to the browser where it is stored as HTML 135 in memory 104. In step 310, the browser determines whether HTML 135 contains any anchor link elements with SOUND attributes. If so, certain initialization steps 312-322 are performed.

Step 312 is the top of a loop performed for each anchor link element in HTML 135 that has the format shown in FIG. 2. If, in step 314, the audio file specified in the anchor link element has not been prefetched, then, in step 316, information about the audio file is stored in a data structure (such as a database or table). (If the audio file has already been prefetched, its URL is already present in the data structure, as discussed below). The stored information includes an identification of the region of the screen where the "playme" information 214 is located, the URL of the audio file, and the current COUNT value.

If the audio file has been prefetched then, in step 318, the old COUNT value for the audio file in the database is replaced by the current COUNT value to indicate that the audio file is referenced on the current page. Steps 320 and 322 represent the loop ends.

FIG. 4 is a flow chart showing steps performed by browser 130 to display a Web page. In step 402, the page is displayed on display device 160 in accordance with HTML 135, in a manner known to persons of ordinary skill in the art. It will be understood that there are other steps (not shown) that are performed by browser 130 to display a Web page and to accept input from the user. Many of these steps are not described herein to enhance the clarity of the example.

In step 404, if there are SOUND attributes present in HTML 135, steps 406-418 are performed. Step 406 is the top of a loop performed for each audio file URL in the data structure that has the current value of COUNT (i.e., for each audio file of the current page that was specified with a SOUND attribute). Step 408 fetches the audio file 147 specified by its URL in the data structure.

Memory 104 of FIG. 1 includes a temporary memory 180 of a predetermined size, such as one megabyte. In step 410, if the fetched audio file will fit in an unused portion of

temporary memory 180, it is stored therein in step 412 and a pointer to its location in temporary memory is placed in the data structure. If the audio file will not fit in temporary memory, steps 410–416 remove audio files of pages viewed the longest time ago from temporary memory until the fetched audio file will fit. If, in step 414, only audio files for the current page remain in temporary memory and there still is not room, the fetched audio file is discarded in step 415 and control passes to step 418. Step 418 represents a loop end.

Steps 420–430 represent a loop performed to receive user input (such as mouse clicks) and to respond thereto. In step 422, if the user clicks on an area link associated with an audio file, then steps 424–428 are performed. In step 424, if the audio file was specified in HTML 135 with a SOUND attribute and if the audio file actually was prefetched and stored in step 412 then, in step 428, the audio is retrieved from the temporary memory and played. Because retrieval from temporary memory does not involve an access over line 106, it occurs very fast and there is a very short period of time between the user's click and the sound. In contrast, if the audio file was not specified in the HTML as prefetchable, or if there was no room in temporary memory to prefetch the audio file, the audio file 147 is fetched over line 106 in step 426 and played. Step 426 may involve for example, an internet access and as a general rule takes longer than step 428.

FIGS. 5(a) through 5(c) provide an example of prefetching audio files. As shown in the "HTML" column, audio files to be prefetched are specified using a SOUND attribute 500. The "Page Display" column indicates that the text graphic associated with each audio file is displayed on the Web page. In FIG. 5(a), the user has viewed two pages. A first page 502 has two instances of a link to an audio file Sound1.au. A second page 504 has a link to an audio file Sound2.au and a link to an audio file Sound3.au. In FIG. 5(a) a database 506 stores region, URL, COUNT value, and a temporary memory location for each audio file. The audio files have been prefetched and stored in temporary memory 508?

In FIG. 5(b), a new page 515 is viewed. The new page also has a link to audio file Sound1.au. Audio file Sound1.au was previously prefetched when page 502 of FIG. 5(a) was displayed. Thus, in accordance with step 318 of FIG. 3, the COUNT value 520 in database 506 for audio file Sound1.au is changed to the current COUNT value of "3" to indicate that the audio file was most recently referenced on third page 515.

In FIG. 5(c), a new page 525 is viewed. The new page has a link to an audio file Sound4.au. In the example, after Sound4.au was fetched over line 106, it was apparent that there was insufficient room in temporary memory 508 for the file. Thus, Sound2.au was removed from temporary memory 508 because it was the first file having a lowest COUNT value ("2") (See FIG. 5(b)). In the example, once Sound2.au was removed, there was room in temporary memory for Sound4.au and that file was stored in temporary memory 506 in accordance with step 412 of FIG. 4. Thus, in FIG. 5(c), audio files Sound1.au, Sound 3.au, and Sound 4.au have been prefetched and can be played immediately when the user selects their displayed links.

Prefetching is preferably performed using some scheme that allows many computer activities to progress in parallel. A preferred embodiment uses the multi-threading capability of Solaris 2.5 is the preferred means, but others can be used too. Thus, if the user issues any commands (e.g., to go to another page) before the download/prefetch is completed,

then the computer will accept this input and perform the requested action.

Any audio file downloads in progress when the user issues a command to go to a new page will be treated as follows:

a) If the user uses a "open in new window" command (that is, the new page will open in a different window from that used to display the current page), then the sound downloads are temporarily suspended while the new page is downloaded and then resumed as soon as its HTML text has been downloaded.

b) If the user uses a "open in same window" command (that is, the new page will replace the current page and be displayed in the same window), then all sound downloads in progress are immediately terminated.

In summary, the present invention allows the designer of a web page to specify whether to prefetch audio files that are to be played when the user selects an associated text or graphic. If no SOUND attribute is present in the HTML, the audio file is not prefetched. It is desirable that the designer can specify which audio files to prefetch, since some audio files are very large. Such files would take too much space in the temporary memory.

Other embodiments will be apparent to those skilled in the art from consideration of the specification and practice of the invention disclosed herein. It is intended that the specification and examples be considered as exemplary only, with a true scope of the invention being indicated by the following claims and equivalents.

What is claimed is:  
1. A method of processing HTML that describes a Web page, comprising the steps performed by a data processing system, of:

receiving HTML describing the Web page, the HTML describing the Web page including at least one SOUND attribute identifying at least one audio file associated with the Web page that should be prefetched;

reviewing the received HTML to locate the at least one SOUND attribute in the HTML;

prefetching automatically the audio file identified by the at least one SOUND attribute when during the reviewing step the SOUND attribute is located in the HTML; and

storing the prefetched audio file in a memory of the data processing system to allow the audio file to be played immediately upon activation of the audio file by a user.

2. The method of claim 1, wherein the storing step includes the step of:

storing a current page number of the Web page in the memory in association with the prefetched audio file.

3. The method of claim 1 wherein the memory of the data processing system includes a temporary memory, wherein the storing step includes the step of storing the prefetched audio file in the temporary memory when there is room in the temporary memory for the prefetched audio file.

4. The method of claim 1 wherein the memory of the data processing system includes a temporary memory, wherein the storing step includes the step of deleting from the temporary memory a second audio file referenced in a least recently used Web page.

5. A method of displaying a Web page, comprising the steps performed by a data processing system, of:

receiving HTML describing the Web page, the HTML containing a SOUND attribute associated with an audio file identifying the audio file associated with the Web page that should be prefetched;

5,903,727

7

reviewing the received HTML to locate the SOUND attribute in the HTML;

prefetching automatically the audio file identified by the SOUND attribute in the HTML when during the reviewing step the SOUND attribute is located in the HTML;

storing the prefetched audio file in a memory of the data processing system to allow the audio file to be played immediately upon activation of the audio file by a user;

displaying the Web page in accordance with the HTML;

determining that a user has activated by the audio file by selecting a portion of the displayed Web page, associated with the audio file; and

playing the prefetched audio file stored in the memory.

6. The method of claim 5, wherein the storing step includes the step of:

storing a current page number of the Web page in the memory in association with the prefetched audio file.

7. The method of claim 5 wherein the memory of the data processing system includes a temporary memory, and wherein the storing step includes the step of storing the prefetched audio file in the temporary memory when there is room in the temporary memory for the prefetched audio file.

8. The method of claim 5 wherein the memory of the data processing system includes a temporary memory, and wherein the storing step includes the step of deleting from the temporary memory a second audio file referenced in a least recently used Web page.

9. The method of claim 5,

wherein the portion of the displayed Web page is linked to the prefetched audio file; and

wherein the determining step includes the step of determining that the user has selected the portion of the displayed Web page linked with the audio file.

10. A method of specifying an audio file on a Web page, comprising the steps, performed by a data processing system, of:

determining a location of graphic data associated with the audio file on the Web page;

storing in a memory of the data processing system an HTML element having the format:

`<A HREF=url SOUND>graphic data </A>`

where the HTML element is stored in the memory in a location associated with the location of the graphic data on the Web page and the format indicates that an audio file associated with the HTML element is to be prefetched automatically when during a review of HTML describing the Web page the HTML element is located in the HTML.

11. A computer program product, comprising:

a computer usable medium having a computer readable program code mechanism embodied therein configured to display a Web page, the computer readable program code mechanism in said program product including:

a computer readable program code mechanism configured to cause a computer to effect receiving HTML describing the Web page, the HTML containing a SOUND attribute associated with an audio file identifying the audio file associated with the Web page that should be prefetched;

a computer readable program code mechanism configured to cause a computer to effect reviewing the received HTML to locate the SOUND attribute in the HTML;

8

a computer readable program code mechanism configured to cause the computer to effect prefetching automatically the audio file identified by the SOUND attribute in the HTML when during the review performed by the computer the SOUND attribute is located in the HTML; and

a computer readable program code mechanism configured to cause the computer to effect storing the prefetched audio file in a memory of the data processing system to allow the audio file to be played immediately upon activation of the audio file by a user.

12. The computer program product of claim 11, wherein the computer readable program code mechanism in said program product further comprises:

a computer readable program code mechanism configured to cause the computer to effect displaying the Web page in accordance with the HTML;

a computer readable program code mechanism configured to cause the computer to effect determining that a user has selected a portion of the displayed Web page associated with the audio file; and

a computer readable program code mechanism configured to cause the computer to effect playing the prefetched audio file stored in the memory.

13. A computer system that processes HTML describing a Web page, the computer system having a memory, comprising:

an HTML receiving portion that receives the HTML describing the Web page, the HTML describing the Web page including at least one SOUND attribute identifying at least one audio file associated with the Web page that should be prefetched;

a reviewing portion that reviews the received HTML to locate the at least one SOUND attribute in the HTML;

a prefetch portion that prefetches automatically the audio file identified by the at least one SOUND attribute when during the review performed by the reviewing portion the SOUND attribute is located in the HTML; and

a storage portion that stores the prefetched audio file in the memory of the computer system to allow the audio file to be played immediately upon activation of the audio file by the user.

14. The computer system of claim 13, wherein the storage portion includes a portion that stores a current page number of the Web page in the memory in association with the prefetched audio file.

15. The computer system of claim 13,

wherein the memory of the computer system includes a temporary memory, and

wherein the storage portion includes a portion that stores the prefetched audio file in the temporary memory when there is room in the temporary memory for the prefetched audio file.

16. The computer system of claim 13,

wherein the memory of the computer system includes a temporary memory,

wherein the storage portion includes a portion that deletes from the temporary memory a second audio file referenced in a least recently used Web page.

17. A computer readable medium carrying one or more sequences of instructions for processing HTML describing a Web page, wherein execution of the one or more sequences of instructions by one or more processors causes the one or more processors to perform the steps of:

5,903,727

9

receiving HTML describing the Web page, the HTML  
describing the Web page including at least one SOUND  
attribute indicating that at least one audio file associ-  
ated with the Web page should be prefetched;  
reviewing the received HTML to locate the at least one 5  
SOUND attribute in the HTML; prefetching automati-  
cally the audio file identified by the at least one sound

10

attribute when during the reviewing step the SOUND  
attribute is located in the HTML; and  
storing the prefetched audio file in a memory of the data  
processing system to allow the audio file to be played  
immediately upon activation of the audio file by a user.

\* \* \* \* \*



**United States Patent** [19]  
**LeMole et al.**

[11] **Patent Number:** **6,009,410**  
[45] **Date of Patent:** **Dec. 28, 1999**

- [54] **METHOD AND SYSTEM FOR PRESENTING CUSTOMIZED ADVERTISING TO A USER ON THE WORLD WIDE WEB**
- [75] Inventors: **Suzanne L. LeMole**, Murray Hill;  
**Steven Howard Nurenberg**,  
Manalapan, both of N.J.; **Joseph  
Thomas O'Neil**, Staten Island, N.Y.;  
**Peter H. Stuntebeck**, Little Silver, N.J.
- [73] Assignee: **AT&T Corporation**, New York, N.Y.
- [21] Appl. No.: **08/951,298**
- [22] Filed: **Oct. 16, 1997**
- [51] **Int. Cl.<sup>6</sup>** ..... **G06F 17/60**
- [52] **U.S. Cl.** ..... **705/14; 705/1; 707/102;  
709/219; 379/201; 348/8**
- [58] **Field of Search** ..... **705/1, 14, 10;  
707/102; 709/219; 379/201; 348/8; 455/4.2**

[56] **References Cited**

**U.S. PATENT DOCUMENTS**

5,661,516	8/1997	Carles	348/8
5,717,923	2/1998	Dedrick	707/102
5,754,938	5/1998	Herz et al.	455/4.2
5,793,972	8/1998	Shane	709/219
5,848,396	12/1998	Gerace	705/10
5,850,433	12/1998	Rondeau	379/201

**OTHER PUBLICATIONS**

DR-Link, ZD Net Free Personalized News on Internet, pp. 1-2, Oct. 13, 1995.

Dr-Link, Companies Eye Lucrative Niche in Targeting ADs, Personalizing Content in Potentially Huge Web Market. Interactive Daily. Nov. 8, 1996, pp. 1-3.

*Primary Examiner*—Allen R. MacDonald  
*Assistant Examiner*—M. Irshadullah

[57] **ABSTRACT**

A customized advertising repository server is connected on the World Wide Web (WWW), which can be accessed by a registered user through his or her browser either by clicking on an icon, or by inputting the specific URL address of the particular server which stores that user's advertising repository. When the user accesses his or her customized ad repository through the browser, a composite advertising page is dynamically configured by the Customized Advertising Repository (CAR) server for that particular user based on that user's previously provided user profile. Furthermore, at least a portion of that composite advertising page can be dynamically configured on a context dependent basis determined from the particular Web site or sites that the user has accessed prior to accessing the CAR. The dynamically configured composite page or pages of advertising provided to the user may contain plural static images, streaming banners, 3-D images, animation, video and/or audio clips, using any of the technologies available on the Web for presenting textual and/or visual information. Such a composite page or pages is configured from a database which stores such images, banners, animation, etc., from plural advertisers. The customized page is created by selecting from among a storehouse of plural different subscribing advertisers and their associated banner ads, images, etc., those particular images, etc. that will be elements of the customized page based on the user's specific areas of interest as determined from the profile, and/or the context dependency. From such dynamically configured composite page or pages, the user can then click on a particular image, video window, banner, etc., to retrieve, through a hyperlink, further information directly from the selected advertiser's own Web site or mirror Web site.

**20 Claims, 4 Drawing Sheets**

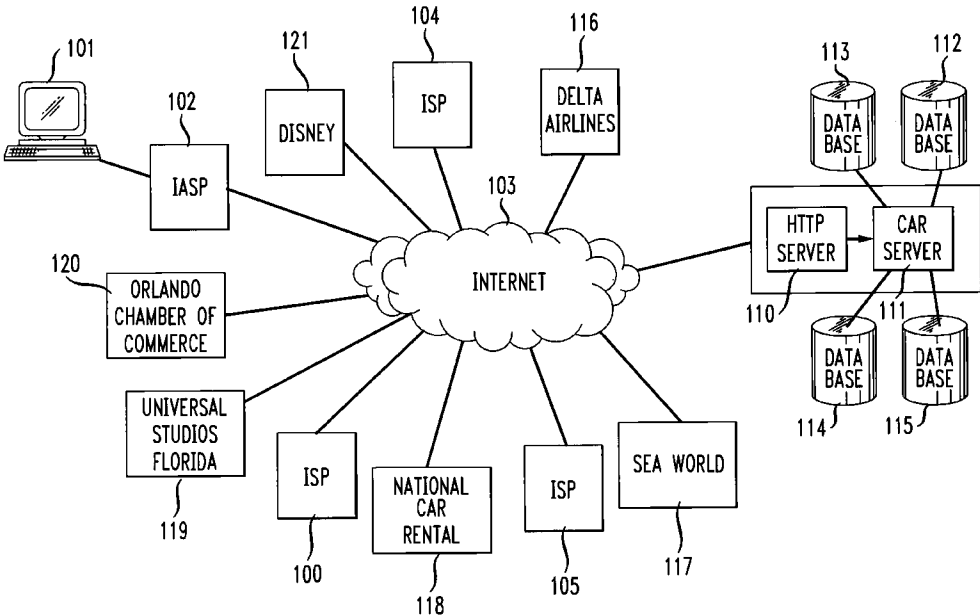




FIG. 1

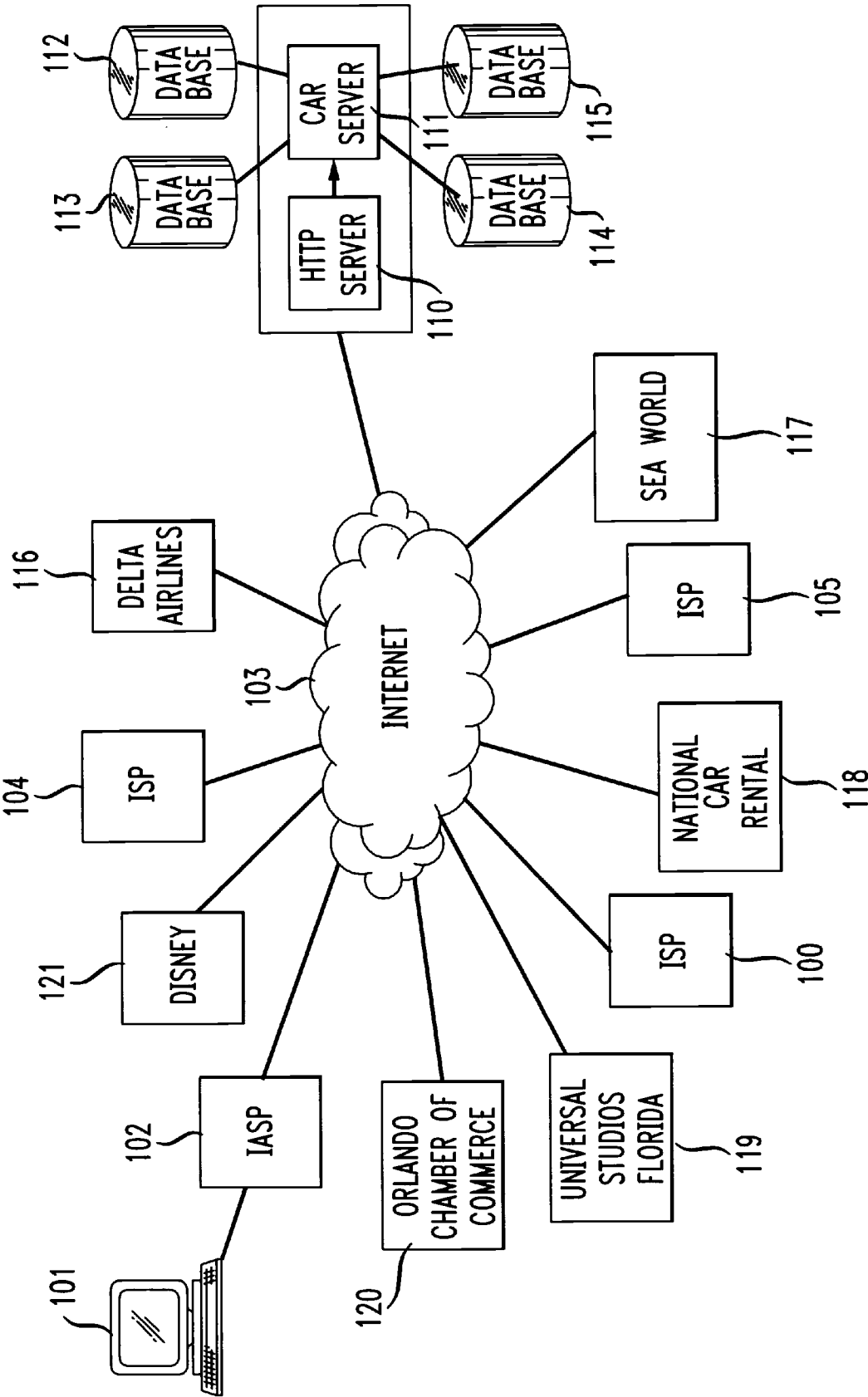


FIG. 2

File Edit View Go Bookmarks Option Directory Window Help

Back

Forward

Home

Reload

Images

Open

Print

Find

Stop

CUSTOMIZED AD REPOSITORY  
REGISTRATION FORM

NAME:

HOME ZIP CODE:

AGE:

INTERESTS:

☐ ART

☐ AUTOMOBILES

☐ COMPUTERS

☐ FASHION

☐ FOOD

☐ KIDS

☐ MOVIES

☐ TELEVISION

☐ TRAVEL

☐ SHOPPING

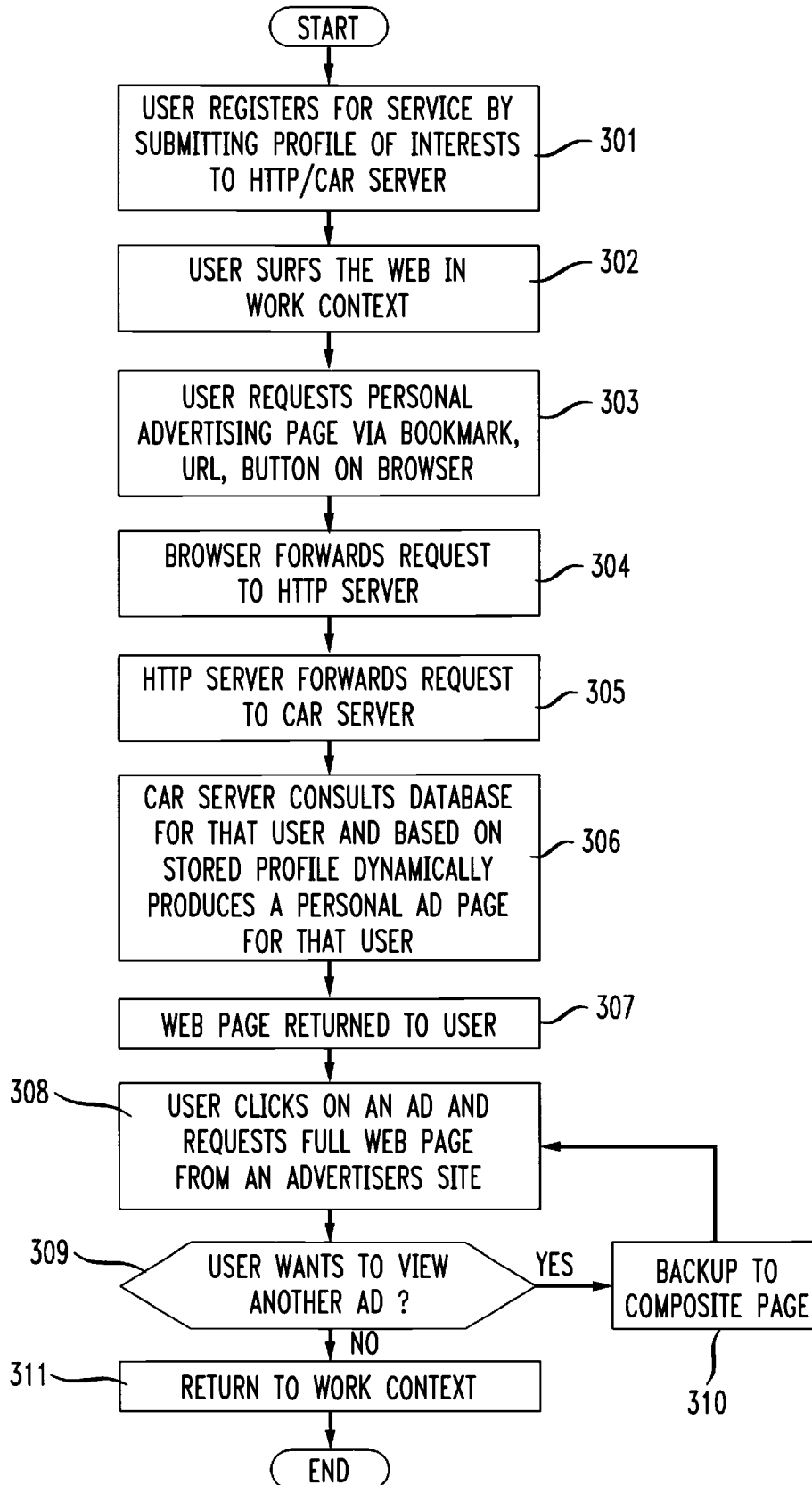
☐ THEATER

☐ SPORTS

SUBMIT



FIG. 3



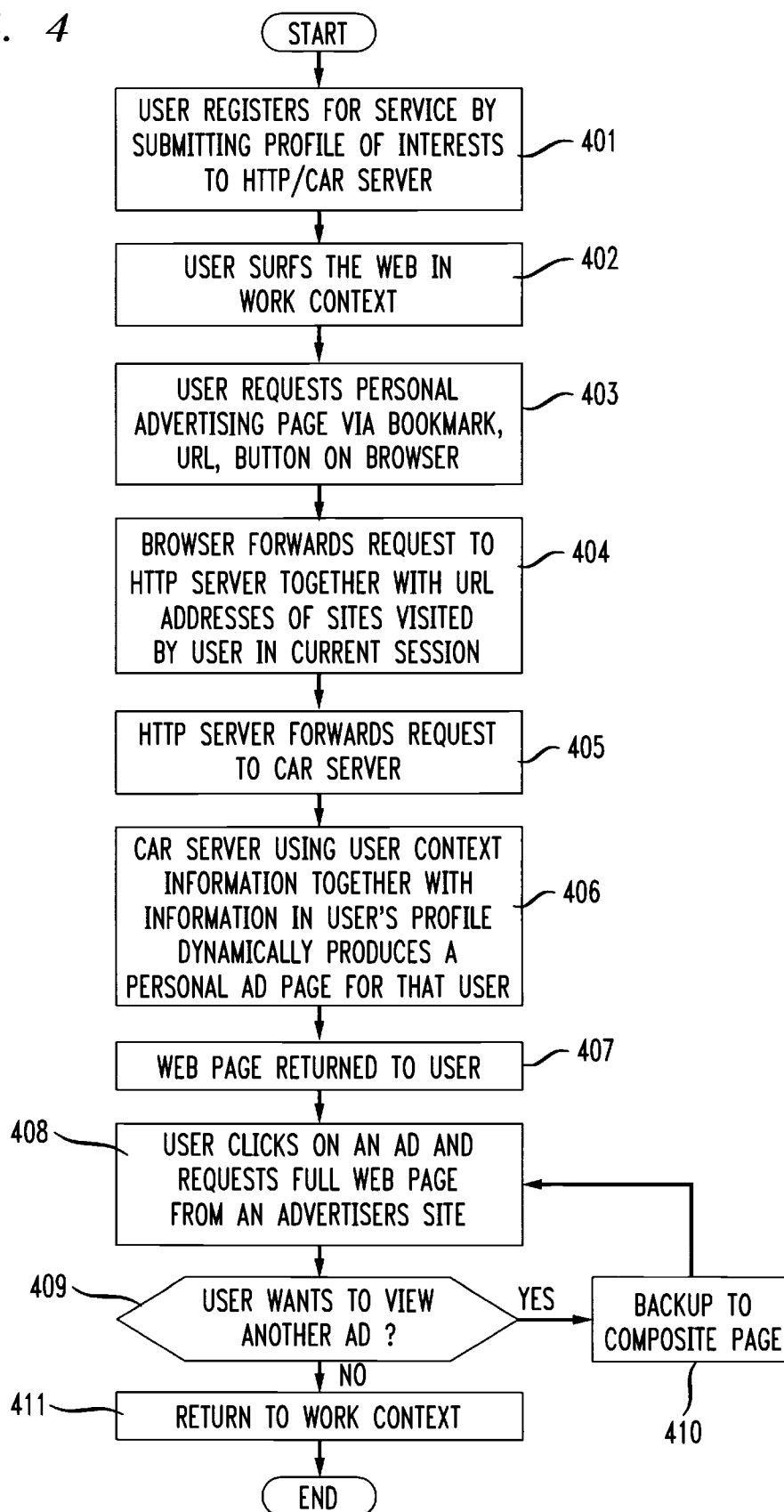
U.S. Patent

Dec. 28, 1999

Sheet 4 of 4

6,009,410

FIG. 4



6,009,410

1

**METHOD AND SYSTEM FOR PRESENTING  
CUSTOMIZED ADVERTISING TO A USER  
ON THE WORLD WIDE WEB**

**TECHNICAL FIELD**

This invention relates to the World Wide Web, and more particularly, to a method and system for presenting customized advertising to a user on the Web.

**BACKGROUND OF THE INVENTION**

As the World Wide Web steadily becomes an increasingly more popular gateway for accessing sources of information, entertainment, shopping, and various interactive services to millions of users around the world, information providers which supply such information, entertainment, goods and interactive services via HTML-formatted pages have taken advantage of the captured nature of the Web users who access its Web site by "selling" part of the "real estate" on its pages to advertisers who may advertise products or services that may or may not be related to the content or nature of the site. Currently, many of the pages provided by information providers contain advertising that takes various forms such as banner ads across the top or bottom of a page. Such ads may include scrolled information containing images that change with time. Disadvantageously, from an advertiser's perspective, Web users have a tendency to mentally "tune-out" such advertising as they read or interact with the information displayed on the main work area of a page. Furthermore, by utilizing a portion of the valuable "real estate" on a Web page for advertising, the remaining available work area on the page is reduced from its maximum full-screen capabilities. Techniques currently also exist for streaming both audio and video across the Internet. Such an audio component can be incorporated as part of the information content of a Web page as well as part of the accompanying advertising, thereby leading to a potential conflict between the audio components of each. Also, since the size of the ad itself on the Web page is limited to only a small portion of the entire page, the full capabilities that could be presented to a Web user through that ad cannot be fully deployed for their maximum visual impact. An example of the latter is holographic 3D experiences that are currently evolving on the Web.

Technologies currently exist which deliver information and advertising to users through a screen saver during periods in which the user is not accessing the Web, or not using his or her terminal for other purposes. The user, however, may not be there to see the information when it is downloaded. Thus, as with a television commercial which more often than not is presented to an empty room or to a disinterested viewer, advertising on the Web as it is currently presented to a user are not likely to achieve the advertiser's desired impact.

**SUMMARY OF THE INVENTION**

In accordance with the present invention, a customized advertising repository server is connected on the World Wide Web (WWW), which can be accessed by a registered user through his or her browser either by clicking on an icon, or by inputting the specific URL address of the particular server which stores that user's advertising repository. When the user is interacting on the Web in a usual manner retrieving information for business or personal reasons, he or she may desire a break from those browsing activities. Although accessing a general advertising site may not be a browsing activity that a user would frequently or even

2

periodically pursue, the ability to view advertising that is specifically geared to that user's individual interests is more likely to be attractive to a user and, with proper incentives, an activity the user will voluntarily pursue. Further, the user may in fact be stimulated to exit his work context mode in which he or she is normally browsing and enter a commercial context mode to view advertising for financial incentives, such as being presented with special offers and/or discounts that are only available to that user through such a customized advertising repository server, or through other incentives, as an example, which tie the accessing of the customized advertising repository with reduced Internet access charges. In accordance with the invention, when the user accesses his or her customized ad repository through the browser, a composite advertising page is dynamically configured by the Customized Advertising Repository (CAR) server for that particular user based on that user's previously provided user profile. Furthermore, at least a portion of that composite advertising page can be dynamically configured on a context dependent basis determined from the particular Web site or sites that the user has accessed prior to entering the commercial context mode. This context dependency that links the advertising presented to the user by the CAR to the Web site or sites previously accessed by the user can be based on key words associated with just the site accessed immediately prior to entering the commercial context mode. Alternatively, the advertising can be linked with key words associated with a plurality of previously accessed sites. The dynamically configured composite page or pages of advertising provided to the user may contain plural static images, streaming banners, 3-D images, animation, video and/or audio clips, using any of the technologies available on the Web for presenting textual and/or visual information. Such a composite page or pages is configured from a database which stores such images, banners, animation, etc., from plural advertisers. The customized page is created by selecting from among a storehouse of plural different subscribing advertisers and their associated banner ads, images, etc., those particular images, etc. that will be elements of the customized page based on the user's specific areas of interest as determined from the profile, and/or the context dependency. From such dynamically configured composite page or pages, the user can then click on a particular image, video window, banner, etc., to retrieve, through a hyperlink, further information directly from the selected advertiser's own Web site or mirror Web site.

Since the dynamically created advertising page or pages that is/are downloaded contain advertising material that is specifically customized to the user's interests, and which are updated regularly by the customized advertising repository server, the user is induced to periodically access his or her customized advertising repository to view the current offers presently being advertised and promoted, and which because of the customization to the individual user have a higher than average probability of being of interest to the user.

**BRIEF DESCRIPTION OF THE DRAWINGS**

FIG. 1 is a block diagram of a telecommunications system in which a user is connected through his or her client terminal to the Internet for accessing various Web sites through the terminal's browser program; and through this browser the user also interacts with an HTTP server and a Customized Advertising Repository (CAR) server, the latter which configures an advertising page that is customized for the user, which page contains advertising information and links to other web sites in accordance with the present invention;

6,009,410

3

FIG. 2 shows an exemplary profile registration form which is downloaded to a user's terminal, completed by the user, and returned to the HTTP server and CAR server for processing and subsequent dynamic formulation of a composite page that is customized for that user to advertise goods and services that are likely to be of interest to that user based on the user's expressed areas of interest and demographic data of that user as provided in the profile;

FIG. 3 is a flowchart of the steps associated with a non-context dependent embodiment of the present invention; and

FIG. 4 is a flowchart of the steps associated with a context dependent embodiment of the present invention.

DETAILED DESCRIPTION

With reference to FIG. 1, a user at a client terminal 101 is connected to an Internet Access Service Provider (IASP) 102 and thence to the Internet 103. The connection of terminal 101 to IASP 102 and Internet 103 can be made in various ways as, for example, through a voice-band modem (not shown) over conventional POTS lines through a Local Exchange Carrier (LEC) (not shown). Alternatively, the connection to the Internet can be made over ISDN facilities, through a cable modem over cable television transmission facilities, or through other wired or wireless facilities. Illustratively, IASP 102 can be an access provider such as AT&T WorldNet<sup>SM</sup> on-line service.

When the user at client terminal 101 specifies the URL address of a desired HTTP-formatted page supplied over the World Wide Web (WWW) from an Information Service Provider (ISP), the browser within client terminal 101 sends a request over the Internet 103 to the identified ISP, ISP 104 for example, for that specific page. The requested page, as identified by the user-inputted URL address, is in turn transmitted in packetized form back over the Internet 103 and through IASP 102 to terminal 101 for display on the user's client terminal 101. By sequentially inputting a series of URL addresses manually through the browser or by clicking the terminal's mouse on a hyperlink on one HTTP-formatted page to a next URL address, or through a combination of both, the user is able to navigate through an essentially endless number of URL-addressed pages of information at ISP 104 and any of the other ISPs, such as ISPs 105 and 106, connected to the Internet. A user can thus literally spend hours "surfing the Net" retrieving information, accessing on-line services, and engaging in on-line transactions. Some of the sites may provide information to the accessing user and may be sponsored by governmental agencies, educational facilities, news providers, etc. Others sites may be provided by various companies that provide information in combination with advertising the goods or services that they offer for on-line or off-line purchase. Examples of the latter may be sponsored by chambers of commerce, magazines, vacation destinations, etc. Even other sites may be purely commercial in nature and associated with a company or an individual which or who is advertising its goods or service as, for example, a movie company advertising its newest motion picture releases, or a major conglomerate such as Disney, which may advertise its theme parks, its motion pictures and TV productions, and the associated products it also sells such as video tapes other "Disney" merchandise. Often, some of these sites provide interactive games geared in combination with self-advertising the provider's products and/or services.

In accordance with the present invention, as the user at client terminal 101 "surfs the Net", he or she at some time

4

decides to exit an information-seeking work context mode to enter a commercial context mode to view advertising of products, services, etc., specifically directed to him or her. As previously noted, an incentive may be given to the user by his or her Internet service provider to "visit" such an advertising repository. Such an incentive may be reduced Internet access charges, or the presentation of a special offer or discount for an advertised product or service only available to "visitors" to a certain site, which offers are not available to the general public. Specifically, in accordance with the present invention, a Content Server 108, comprising an HTTP Server 110 and an associated CAR server 111, is connected to the Internet 103 and is accessed by the user at client terminal 101 through that client terminal's browser program either by inputting the URL address of the Content Server 108 manually, through a bookmark, or by clicking on a special icon presented on the user's display by a browser that has been modified to display such an icon. When Content Server 108 receives a request from the user, the HTTP server 110 within recognizes the user either through the input of an ID by the user, or through a "cookie" previously provided to the user's browser by server 110 in a previous interaction. The received request is passed by HTTP server 110 to CAR server 111, which accesses an associated and cooperatively connected database 112 to dynamically configure an advertising page specifically for that user. In order to dynamically configure such a customized advertising page for the user, database 112 stores an electronic profile for each registered user. Such a profile indicates subject areas of interest of each user and a user's demographic data from which information a plurality of images, banners, video clips, sound clips, etc. from different advertisers are combined by CAR server 111 into an aggregated advertising page or pages with hyperlinks to the advertising sites of each of the combined advertisers.

Such a profile of interests and demographic data is provided by the user on-line when the user registers to have access to the customized advertising repository service or off-line through a slow-mail registration process. When registering on-line for first time for the service through HTTP server 110, a profile page, such as illustrated in FIG. 2 is returned to client terminal 101. By using the mouse to electronically check his or her interests, such as travel, sports, etc., and by inputting demographic information such one's marital state, age, number of children, their ages, etc., as well as other information, that user's profile is stored in user profile database 112. Using well known techniques for targeting advertising to audiences based on their stated interests and demographic data, particular advertisers of products and/or services from among those product and/or service providers who subscribe to the service as advertisers are selected by CAR server 111 from an associated subscribing advertisers database 113 to dynamically create a composite advertising page specifically for that individual user based on that user's profile stored in user profile database 112. Each time the user enters the commercial context mode a dynamically created and customized composite HTML-formatted page is then presented to him or her.

As an example, if a user indicates that he is married, has two children ages 12 and 10, is interested in travel and theme parks, and lives in New Jersey, a customized advertising page may at one time be configured comprising ads, banners, etc., for Disney World advertising their 25<sup>th</sup> Anniversary celebration, Delta Airlines advertising special rates to Internet users for trips to Orlando, Fla., Sea World, National Car Rental, Universal Studios, etc. On such a aggregate HTML-formatted page, hyperlinks are provided

6,009,410

5

to each advertiser's individual site on the Internet. Thus, such individual advertiser's sites, **116–121**, are accessible by the user by clicking on the image, banner, video clip, etc., on the composite page. When that same user accesses his customized advertising repository at a subsequent time, the page presented may comprise a composite of different individual advertisers based on other profile and demographic information of that same user. Thus, as another example, if the same user has indicated in his or her profile that he or she was also interested in new computer technology, a composite advertising page is presented to the user upon a subsequent visit to the CAR server that combines banners, images, etc., associated with a new computer magazine, an on-line computer hardware and/or software merchant, an upcoming computer convention in the user's local area, etc. Furthermore, the CAR server **111** may produce and store a record in database **112** associated with the user's profile that notes the combination of advertisers that were previously presented to the user. Thus, at each new visit to the customized advertising repository, the user will be presented with a different and new set of advertisers and/or special offers that was not previously presented.

It has heretofore been assumed that the composite advertising page configured by CAR server **111** comprises separate advertisements for products and/or services determined solely by the user's previously provided profile information stored in user profile database **112**. By providing information relating to the Web sites visited by the user during the work context mode prior to entering the commercial context mode, at least some of the composite images, video clips, audio clips, etc. in the dynamically configured advertising page presented to the user upon entering the commercial mode can be associated with advertisers which have a nexus with the user's previously visited Web site(s). Thus, by modifying the browser program in client terminal **101** to provide information relating to the sites previously visited by the user to HTTP server **110** and CAR server **111**, CAR server **111** will retrieve from the associated subscribing advertiser's database **113** the necessary images, etc., for dynamically formulating a composite page that may be in whole or in part determined by the subject matter content of such previously visited sites. By modifying the browser program in client terminal **101** to accumulate information relating to the user's browsing activities (i.e., the sites visited), such information is forwarded to HTTP server **110** and CAR server **111** when the user makes the request to enter the context mode by clicking the commercial icon, or enters the URL address of HTTP server **110**. Thus, when entering the commercial context, the URL addresses of the last N sites visited by the user are provided together with the request to HTTP server **110**. These URL addresses are translated by a URL to Subject Mapping database **114** into a set of keywords. A Subject to Advertisement Mapping database **115** is then used to determine which advertising is relevant for a particular subject area. When CAR server **111** receives a request from the user's browser to view advertising, the URLs of the previous N sites visited are converted to key words by database **114**, and those key words are then used by database **115** to determine the appropriate associated advertisements to be retrieved from advertising database **113** and configured by CAR server **111** into a composite page that is formulated for the user. Alternatively, the browser program in client terminal **101** can directly access a URL to Subject Mapping database located elsewhere on the Internet to determine key words from each visited site. These key words are then passed together with the request to HTTP server **110** and CAR

6

server **111** for accessing the relevant advertising from Subject to Advertisement Mapping database **115**. For either arrangement, once the relevant advertising is determined from a Subject to Advertisement Mapping database, CAR server **111** determines which particular advertising images, banners, clips, etc., are to be associated with the user's previously visited Web sites, and incorporates these in the dynamically configured composite advertising page. Thus, for example, if the user has previously visited a site related to theater news, such as Playbill On-Line, the dynamically configured composite advertising page may include advertisements for specific Broadway plays or musicals. Further, by combining the context information of the previously visited site or sites with the context independent profile information, such as a user's interest in music, the dynamically configured composite advertising page may include advertisements for specific plays and musicals as well as advertisement by a record company promoting upcoming or new CD releases of music from such musicals.

As described hereinabove, when the user enters the commercial mode by clicking on the icon on his browser or by inputting the URL address of the HTTP server **110**, the CAR server **111** then dynamically configures a composite HTML-formatted advertising page for that user and delivers it over the Internet **103** to the user's client terminal **101**. As an alternative, such composite advertising pages can be configured before the user enters the commercial mode and, using "push" technology, transmitted over the Internet to client terminal **101** for storage within a cache to be immediately ready for display to that user as soon as he or she enters the commercial mode. This can be arranged for either the non-context dependent embodiment or the context dependent arrangement described above. For the latter, the browser program in client terminal **101** periodically sends to HTTP server **110** the URL addresses of the sites accessed by the user through the browser program. Based on some sub-set of those sites and/or the user's profile, CAR server **111** dynamically configures a composite page, and pushes it to the browser program where it is cached for later retrieval by the user when he or she enters the commercial mode. As a further extension, the CAR server **111** can dynamically configure a plurality of different composite pages for the user, which are each pushed over the Internet to the user's client terminal where each is cached for later retrieval by the user.

FIG. **3** illustrates the steps of the present invention for the non-context dependent embodiment described above. At step **301**, the user registers for the service by submitting a profile of his or her interests to the HTTP server. At some later time, at step **302**, the user begins to "surf the Web" in a work context. During the user's session, at step **303**, a request is made through the user's browser for the individualized personal advertising page by entering the URL address of the HTTP server, manually or through a bookmark, or by clicking on a special icon button on the browser. At step **304**, the browser forwards the user's request to the HTTP server. At step **305**, the HTTP server forwards the received request to the CAR server. At step **306**, the CAR server consults its associated database for the profile record associated with the user and, based on the stored profile, dynamically produces a personalized composite advertising page for the user. At step **307**, the composite advertising page is returned to the user's browser. At step **308**, the user clicks on an ad in the composite page to request a full Web page from the requested advertiser's own Web site. At step **309**, the user decides whether to view another ad. If yes, at step **310**, the browser is backed up to



6,009,410

7

the composite page, and the process returns to step 308. If the user, at decision step 309, does not want to view another ad from the composite page, he or she returns to the work context by entering the URL address to the next site he or she wants to access.

FIG. 4 illustrates the steps of the present invention for the context dependent embodiment described above. As in the context independent embodiment of FIG. 3, the user, at step 401, registers for the service by submitting a profile of his or her interests to the HTTP server. At some later time, at step 402, the user begins to "surf the Web" in a work context. During the user's session, at step 403, a request is made through the user's browser for the individualized personal advertising page by entering the URL address of the HTTP server, manually or through a bookmark, or by clicking on a special icon button on the browser. At step 404, the browser forwards the user's request to the HTTP server together with the URL addresses of sites previously visited by the user in the current session. At step 405, the HTTP server forwards the received request to the CAR server. At step 406, the CAR server consults its associated database for the profile record associated with the user and, based on the stored profile, together with the context information associated with the previously visited sites, dynamically produces a personalized composite advertising page for the user. At step 407, the composite advertising page is returned to the user's browser. At step 408, the user clicks on an ad in the composite page to request a full Web page from the requested advertiser's own Web site. At decision step 409, the user decides whether to view another ad. If yes, at step 410, the browser is backed up to the composite page, and the process returns to step 408. If the user, at step 409, does not want to view another ad from the composite page, he or she returns to the work context by entering the URL address to the next site he or she wants to access.

Although the invention has been described in conjunction with providing commercial advertising to the user, the invention can also be applied to providing to the user any other type of information elements when the user exits a first work context mode and enters a second mode of work or entertainment. Thus the information elements presented that are dynamically composited, arranged and presented specifically for the user when the user exits the first work mode may be determined based on other characteristics of the user, such as the user's work habits, the user's financial investments, the user's entertainment choices, an alternate work project of the user, etc., and the banners, icons, etc., that make up the composite page do not necessarily have to relate to the advertisement of any product or service to the user. Rather, the information elements which are presented on a composite page are dynamically configured and presented expressly for that user based on some predetermined information relating to the user and/or the subject matter of those sites on the Internet that that user has previously visited.

The above-described embodiments are illustrative of the principles of the present invention. Other embodiments could be devised by those skilled in the art without departing from the spirit and scope of the present invention.

The invention claimed is:

1. A method of presenting a page of information for delivery over a packetized computer network to a user's browser program running on a client terminal comprising:

- receiving a request for the page made from the user's browser program;
- identifying the user making the request;

8

dynamically forming a composite page of information containing a plurality of information elements and associated hyperlinks to a plurality of other pages on the packetized computer network selected from a repository of a larger plurality of information elements and associated hyperlinks, the information elements and associated hyperlinks selected for the composite page being determined at least in part based on a stored previously provided profile associated with the identified user; receiving from the user's browser program a URL address of at least one site that the user's browser program visited before the request for the page is made, at least one information element and its associated hyperlink on the composite page being determined at least in part by subject matter content of the information at the URL address of the at least one site the user's browser program visited; and

sending the composite page of information elements to the user's client terminal over the packetized computer network.

2. The method of claim 1 wherein the stored profile comprises subject topics of interest to the user.

3. The method of claim 2 wherein the stored profile further comprises demographic information associated with the user.

4. The method of claim 1 wherein at least one of the plurality of information elements on the composite page of information is associated with an advertised product or service and its associated hyperlink is to a URL address of a site on the computer network that is associated with a provider of the advertised product or service.

5. The method of claim 4 wherein the advertisement for a product or service is displayed using one or more of the following presentational techniques: a static banner, a streaming banner, a static image, an icon, a multidimensional image, a video clip, and an audio clip.

6. The method of claim 1 wherein the packetized data network is the Internet.

7. The method of claim 1 further comprising:

associating the URL address of the at least one visited site with at least one keyword related to the information content of the visited site;

associating the at least one keyword with at least one information element and associated hyperlink in the larger repository of information elements and associated hyperlinks; and

selecting for inclusion in the composite page the information element associated with the at least one keyword.

8. A method of providing customized advertising to a user of a client terminal that is connected to the Internet and is running an Internet browser program comprising:

receiving a request from the browser program to view a page of advertising;

identifying the user;

dynamically configuring a composite page of advertising comprising a plurality of advertising images and associated hyperlinks to sites at other URL addresses on the Internet selected from a repository of a larger plurality of advertising images and associated hyperlinks, the advertising images and associated hyperlinks selected for the composite page being determined at least in part based on a stored previously provided profile associated with the identified user;

receiving from the user's browser program a URL address of at least one site that the user's browser program has

9

visited before the request for the advertising page is made, at least one advertising image and its associated hyperlink on the composite page being determined at least in part by subject matter content associated with the information at the URL address of the at least one site the user's browser program visited; and  
5 sending the composite page of advertising via the Internet to the browser program running on the user's client terminal.  
9. The method of claim 8 further comprising:  
10 receiving the profile from the identified user; and  
storing the profile.  
10. The method of claim 8 wherein the stored profile comprises subject topics of interest to the user.  
11. The method of claim 11 wherein the stored profile comprises demographic information associated with the user.  
12. The method of claim 8 wherein the composite advertising page is configured based on the user's profile and is pushed to the user's browser program over the Internet for storage in a cache before the request is received.  
13. The method of claim 8 wherein a plurality of composite advertising pages are configured based on the user's profile and pushed to the user's browser program over the Internet for storage in a cache before the request is received.  
14. The method of claim 8 wherein the hyperlink associated with at least one of advertising image on the composite page is to a URL address of a site on the Internet of a product or service provider associated with the at least one advertising image.  
15. The method of claim 8 wherein the advertising image uses one or more of the following presentational techniques: a static banner, a streaming banner, a static image, an icon, a multi-dimensional image, a video clip, and an audio clip.  
16. The method of claim 9 further comprising:  
17 associating the URL address of the at least one visited site with at least one keyword related to the information content of the visited site;  
18 associating the at least one keyword with at least one advertising image and associated hyperlink in the larger repository of advertising images and associated hyperlinks; and  
19 selecting for inclusion in the composite page the advertising image associated with the at least one keyword.  
20 17. A customized advertising server connected to the Internet comprising:

10

means for receiving a request from a user's browser program to view a page of advertising;  
means for identifying the user making the request;  
a profile database for storing a profile of the user;  
a database repository of a plurality of advertising images and associated hyperlinks;  
means for dynamically configuring a composite page of advertising comprising a plurality of advertising images and associated hyperlinks to sites at other URL addresses on the Internet selected from the database repository of advertising images and associated hyperlinks, the advertising images and associated hyperlinks selected for the composite page being determined at least in part based on the user's profile stored in the profile database; and  
means for receiving from the user's browser program a URL address of at least one site that the user's browser program has visited before the request for the advertising page is made, the means for dynamically configuring a composite page selecting for inclusion on the composite page at least one advertising image and its associated hyperlink that is determined at least in part by subject matter content associated with information at the URL address of the at least one site the user's browser program visited.  
18. The server of claim 17 wherein the user's stored profile comprises subject topics of interest to the interest to the user.  
19. The server of claim 18 wherein the user's stored profile comprises demographic information associated with the user.  
20. The server of claim 19 further comprising:  
a URL-to-keyword database for associating the URL address of the at least one visited site with at least one keyword related to the information content of the visited site;  
a keyword-to-advertising database for associating the at least one keyword with at least one advertising image and associated hyperlink in the larger repository of advertising images and associated hyperlinks; and  
means for selecting for inclusion in the composite page the advertising image associated with the at least one keyword.

\* \* \* \* \*





US006112240A

**United States Patent** [19][11] **Patent Number:** **6,112,240****Pogue et al.**[45] **Date of Patent:** **Aug. 29, 2000**[54] **WEB SITE CLIENT INFORMATION TRACKER**

[75] Inventors: **Michael Alan Pogue**; **Laura Allison Werner**, both of Sunnyvale; **Ralf I. Pfeiffer**, Cupertino; **Pratima Gupta**, Sunnyvale; **Yong Zhang**, Fremont; **James Andrew Clark**, Cupertino, all of Calif.

[73] Assignee: **International Business Machines Corporation**, Armonk, N.Y.

[21] Appl. No.: **08/923,013**

[22] Filed: **Sep. 3, 1997**

[51] **Int. Cl.**<sup>7</sup> ..... **G06F 13/14**; G06F 15/16; G06F 3/00

[52] **U.S. Cl.** ..... **709/224**; 707/513; 709/219; 709/202; 709/203

[58] **Field of Search** ..... 707/1, 10, 513, 707/501; 395/200.33, 200.57, 200.32; 709/203, 202, 227, 300, 224, 219

[56] **References Cited****U.S. PATENT DOCUMENTS**

5,623,656	4/1997	Lyons	707/10
5,712,979	1/1998	Graber et al.	709/224
5,717,860	2/1998	Graber et al.	709/227
5,721,908	2/1998	Lagarde et al.	707/10
5,742,762	4/1998	Scholl et al.	709/223
5,751,956	5/1998	Kirsch	709/203
5,774,670	6/1998	Montulli	709/227
5,793,966	8/1998	Amstein et al.	709/203
5,796,952	8/1998	Davis et al.	709/224
5,812,769	9/1998	Graber et al.	709/228
5,819,285	10/1998	Damico et al.	707/104

**OTHER PUBLICATIONS**

Lee, Lydia; "Beyond the Hits: Mining Web Sites for Traffic Data"; New Media Magazine, Nov. 1996.

Lee, Lydia; "Site Analysis Tools Get Serious"; New Media Magazine, Nov. 1996.

Pogue et al; "A Method for Improved Control of Web Browser Caching Behavior", 1997.

Pogue et al; "System for Tracking Website Usage", 1997.

Lemay, Laura; "Teach Yourself Java in 21 Days"; ISBN: 1-57521-030-4; Sams.net Publishing; Chapters 8 and 14, Dec. 1995.

Kruse, Matt; "Using Server-side includes"; Dr. Dobb's Journal; M&T Publishing; v21, n2, p52(3), Feb. 1996.

Staten, James; "Navigator tricks raise concerns"; MacWeek; Ziff-Davis Publishing; v10, n11, p18(2), Mar. 1996.

Stark, Thom; "Server Side Includes take hold"; LAN Times; Mc-Graw-Hill Inc.; v13, n5, p94(1), Mar. 1996.

Gellman, Robert; "They could be monitoring your every Web move"; Government Computer News; Cahners Publishing Assoc.; v15, n9, p25(1), Apr. 1996.

Linthicum, David S.; "Application development tackles the intranet"; Datamation; Cahners Publishing Assoc.; v42, n15, p113(4), Sep. 1996.

Lemay, Laura; "Teach Yourself Web Publishing with HTML 3.2 in a week", Third Edition; ISBN: 1-57521-192-0; Sams.net Publishing; pp. 439-442, Dec. 1996.

Mendelson, Edward; Java: Create your own applets; PC Magazine; Ziff-Davis Publishing Co.; v16, n11, p141(7), Jun. 1997.

*Primary Examiner*—Mark H. Rinehart

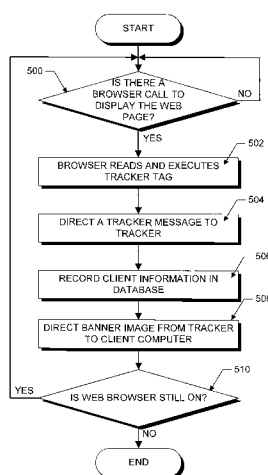
*Assistant Examiner*—Marc D. Thompson

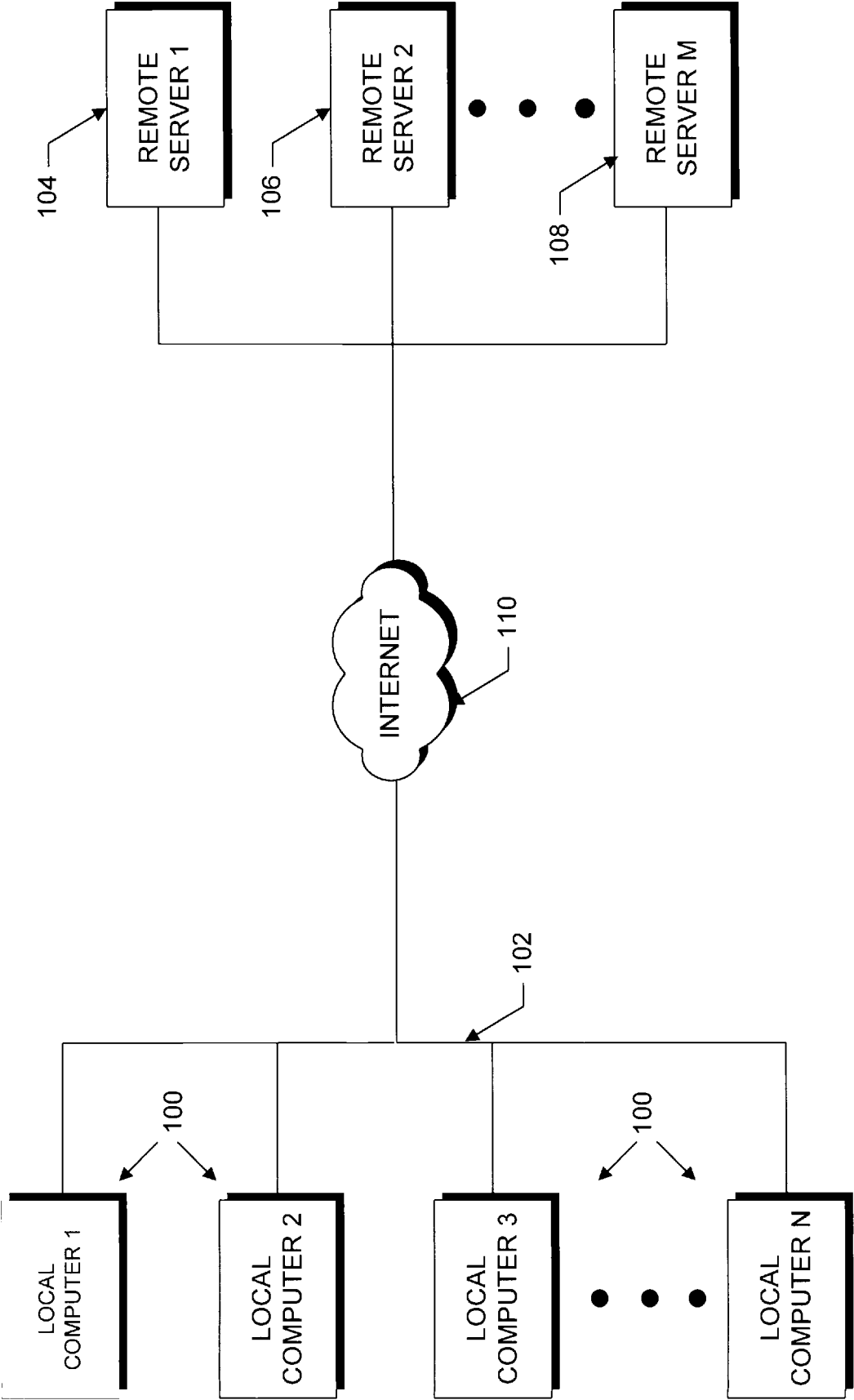
*Attorney, Agent, or Firm*—Kudirka & Jobse, LLP

[57]

**ABSTRACT**

A method and apparatus for obtaining client information relating to a web page in a World Wide Web site utilizes a tracker tag in the code of the web page for initiating a client information tracking program. The tracking program may be on a client computer that is accessing the web page, or a tracking computer that is remote from the client computer. The tracking program is initiated by a tracker message transmitted from a web browser on the client computer to the tracking program when the tracker tag is read by the web browser. In one embodiment, the tracking program first obtains the client information, and then stores the client information in the memory of a computer having the tracking program.

**48 Claims, 7 Drawing Sheets**



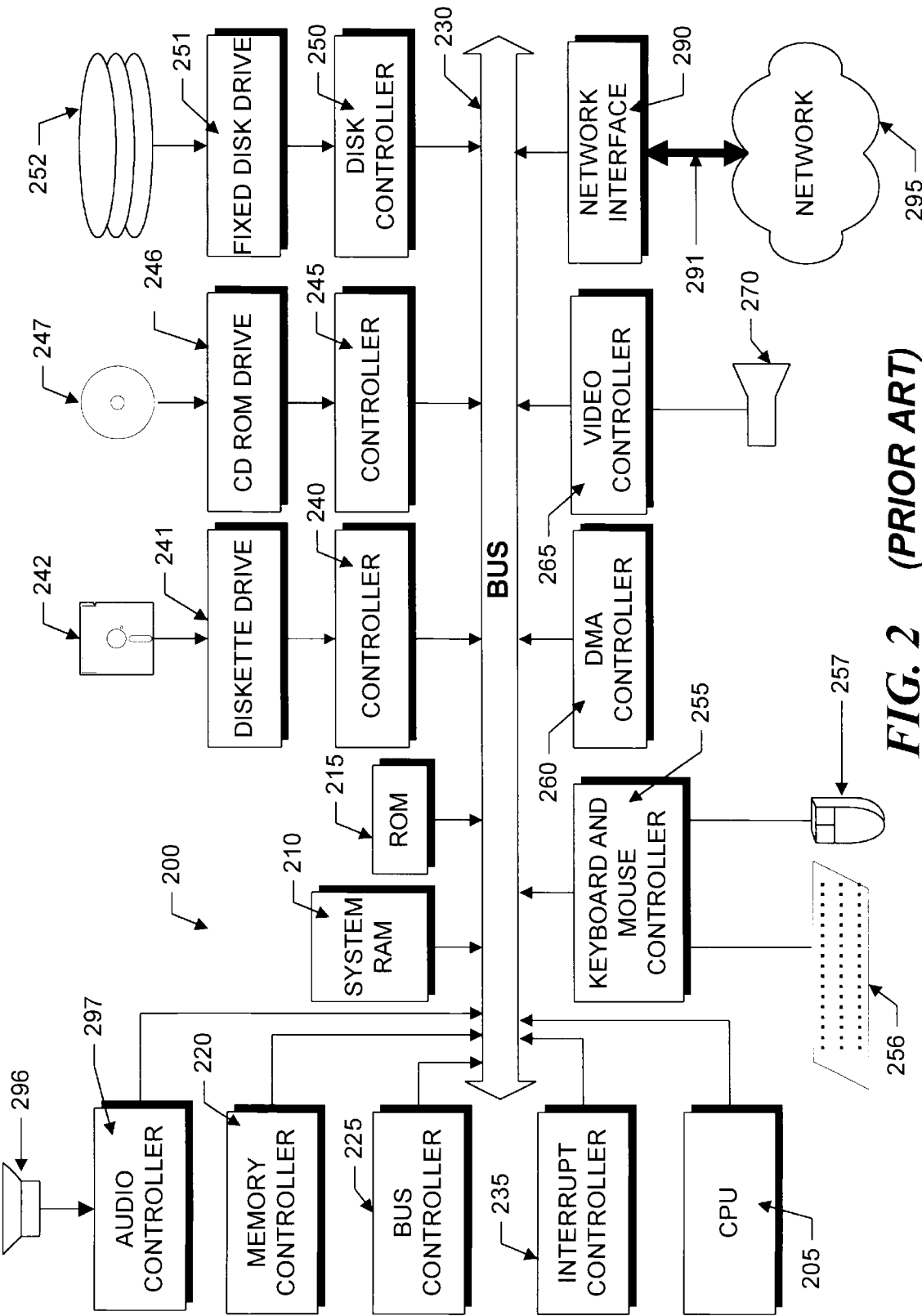


FIG. 2 (PRIOR ART)

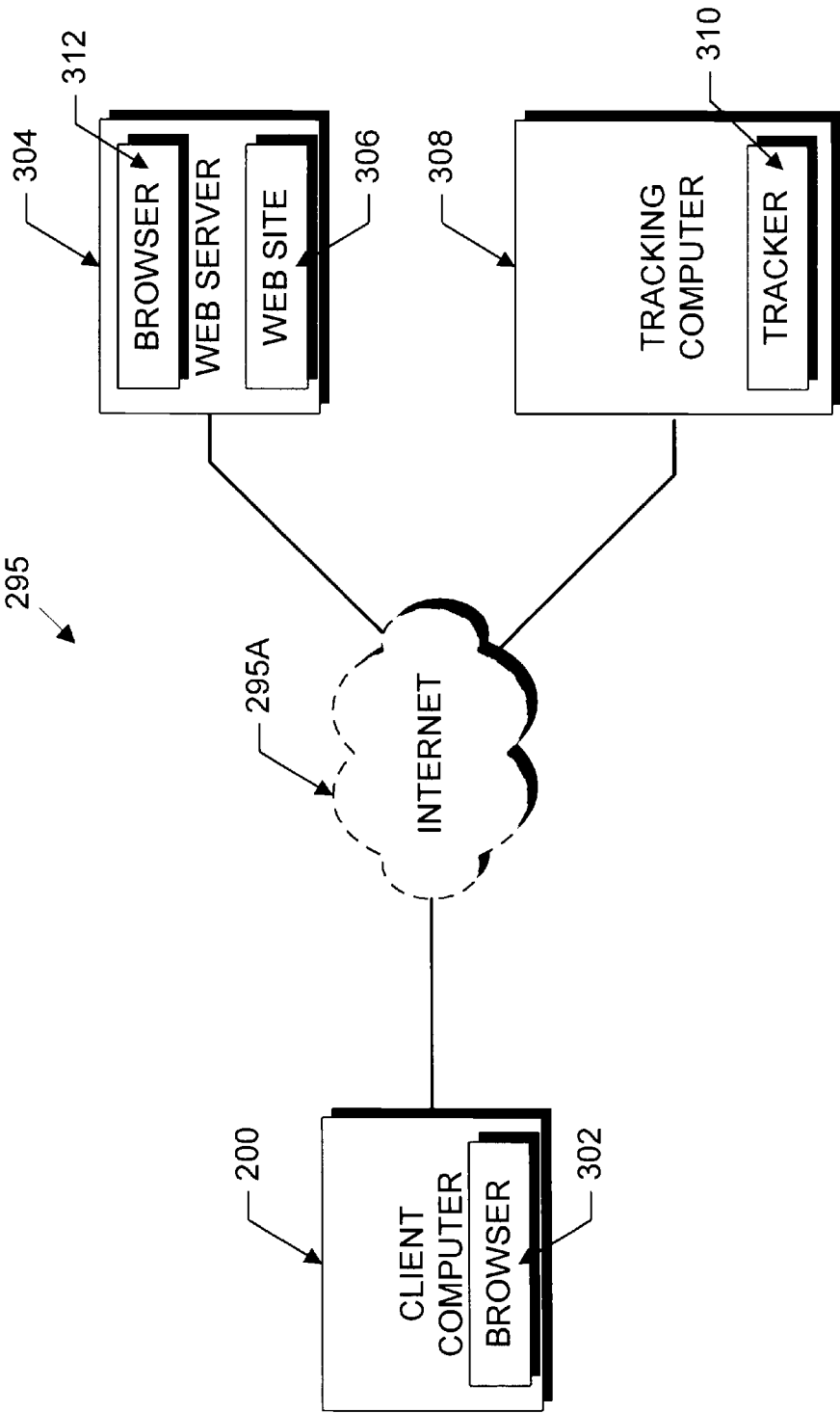
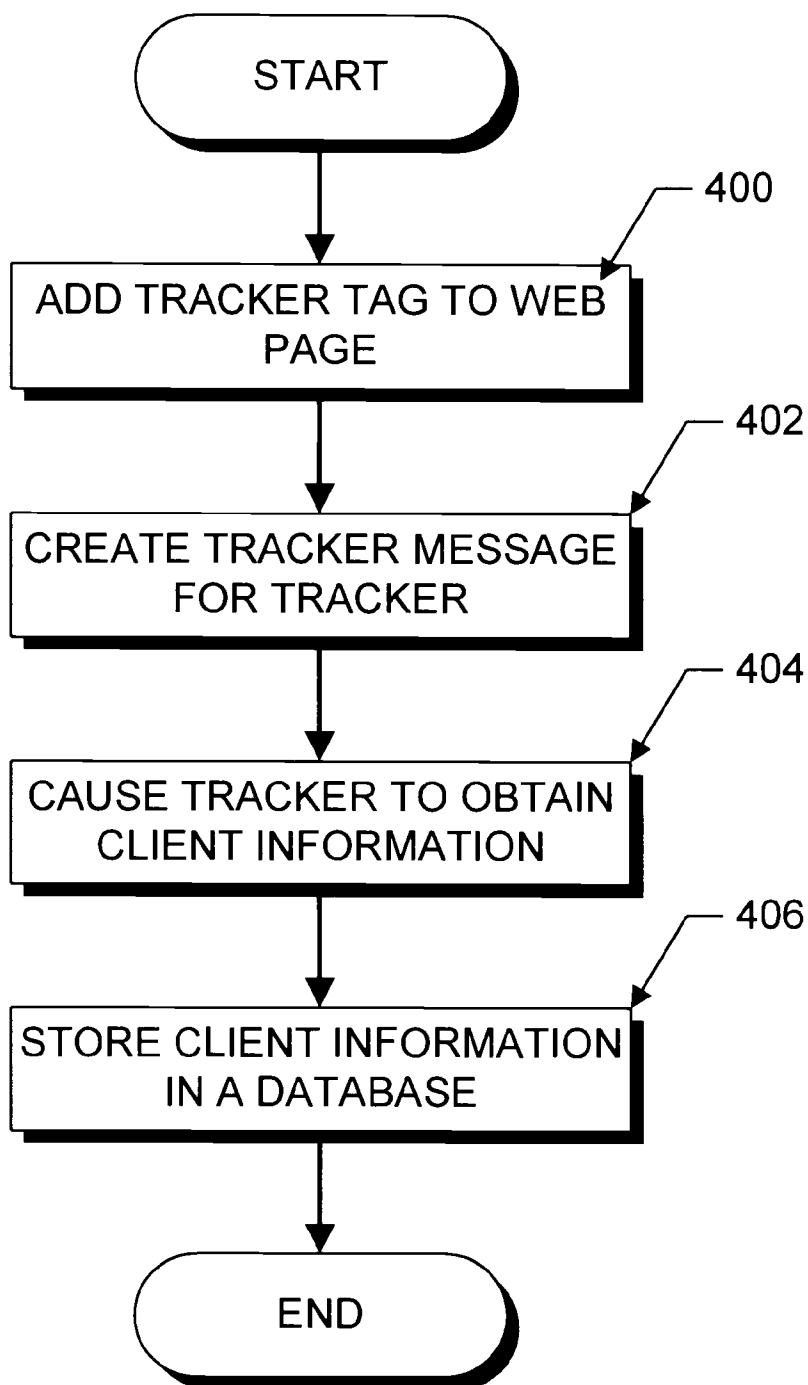
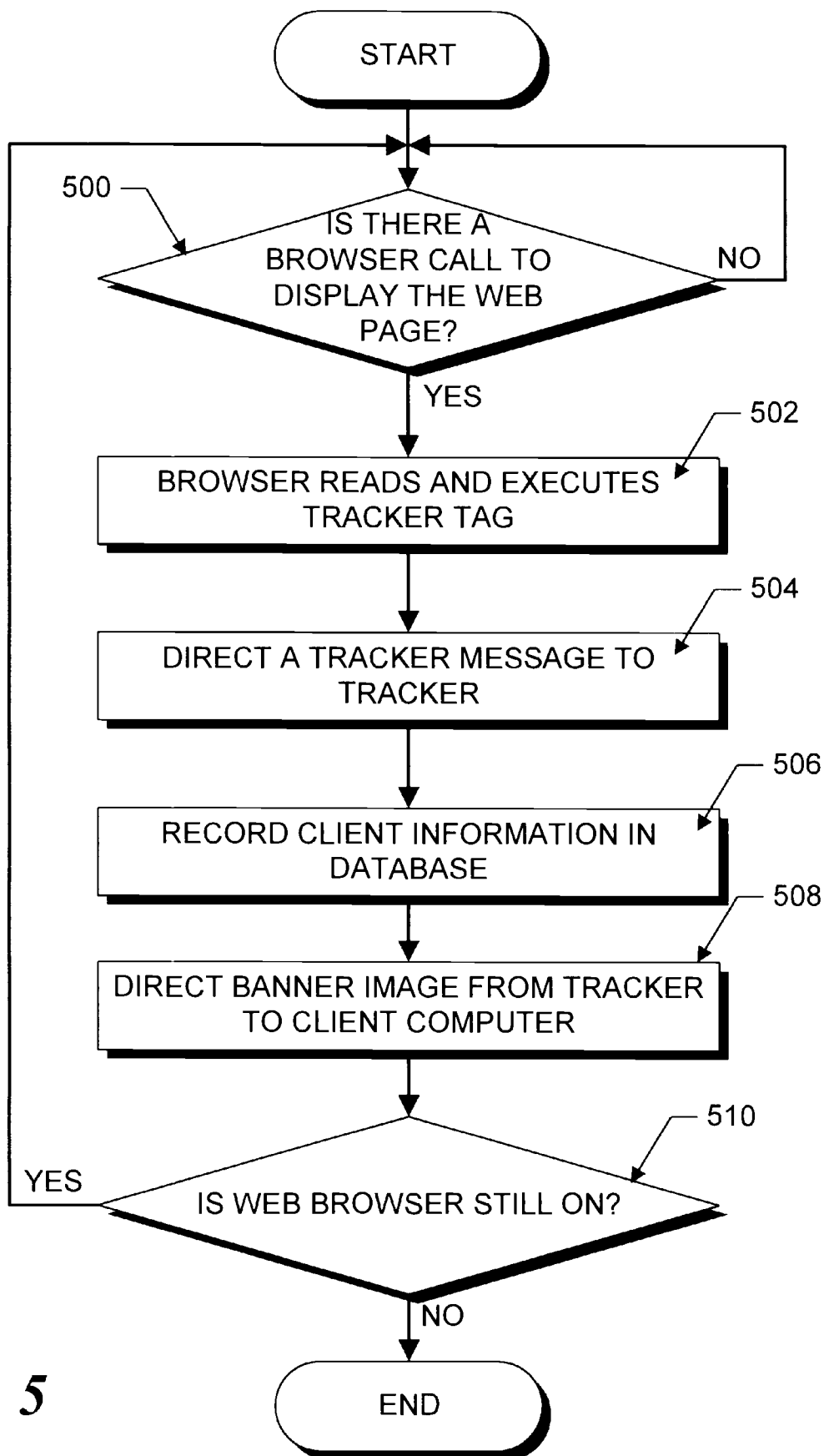
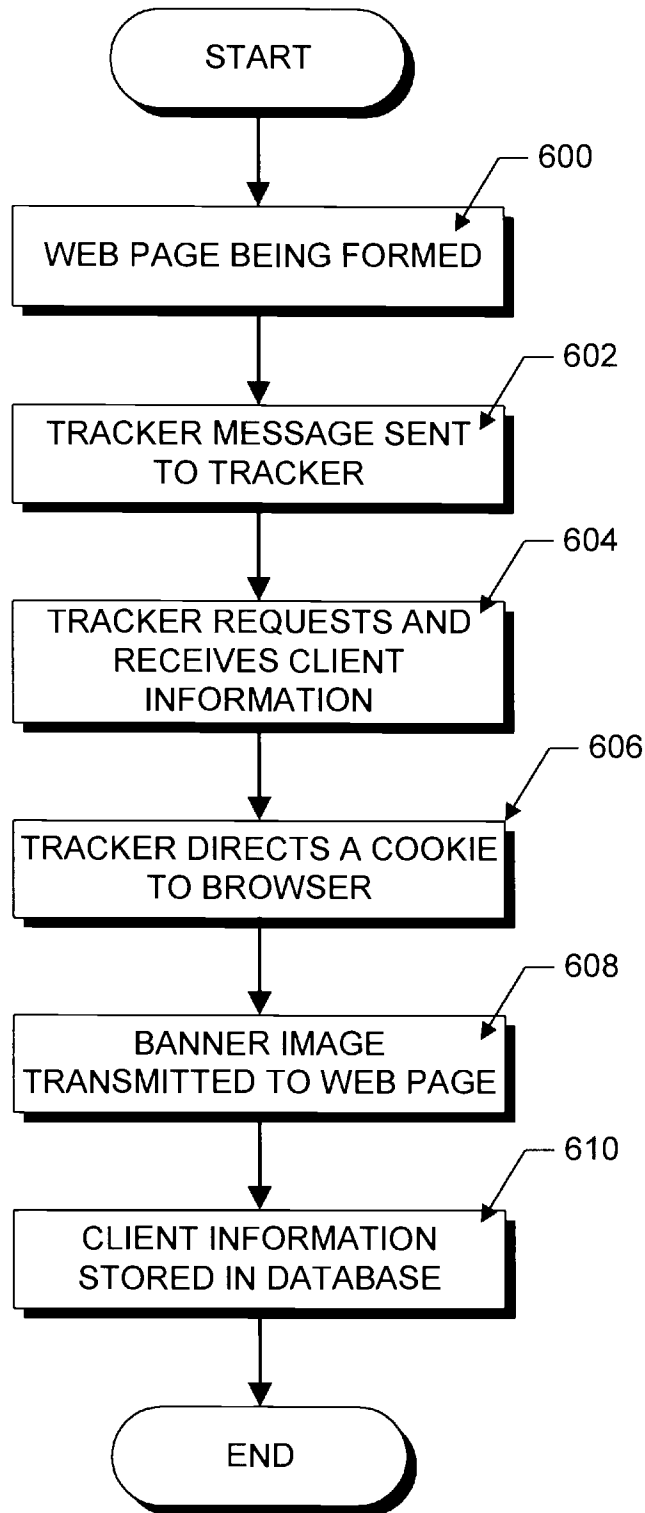


FIG. 3



**FIG. 4**

**FIG. 5**



**FIG. 6**

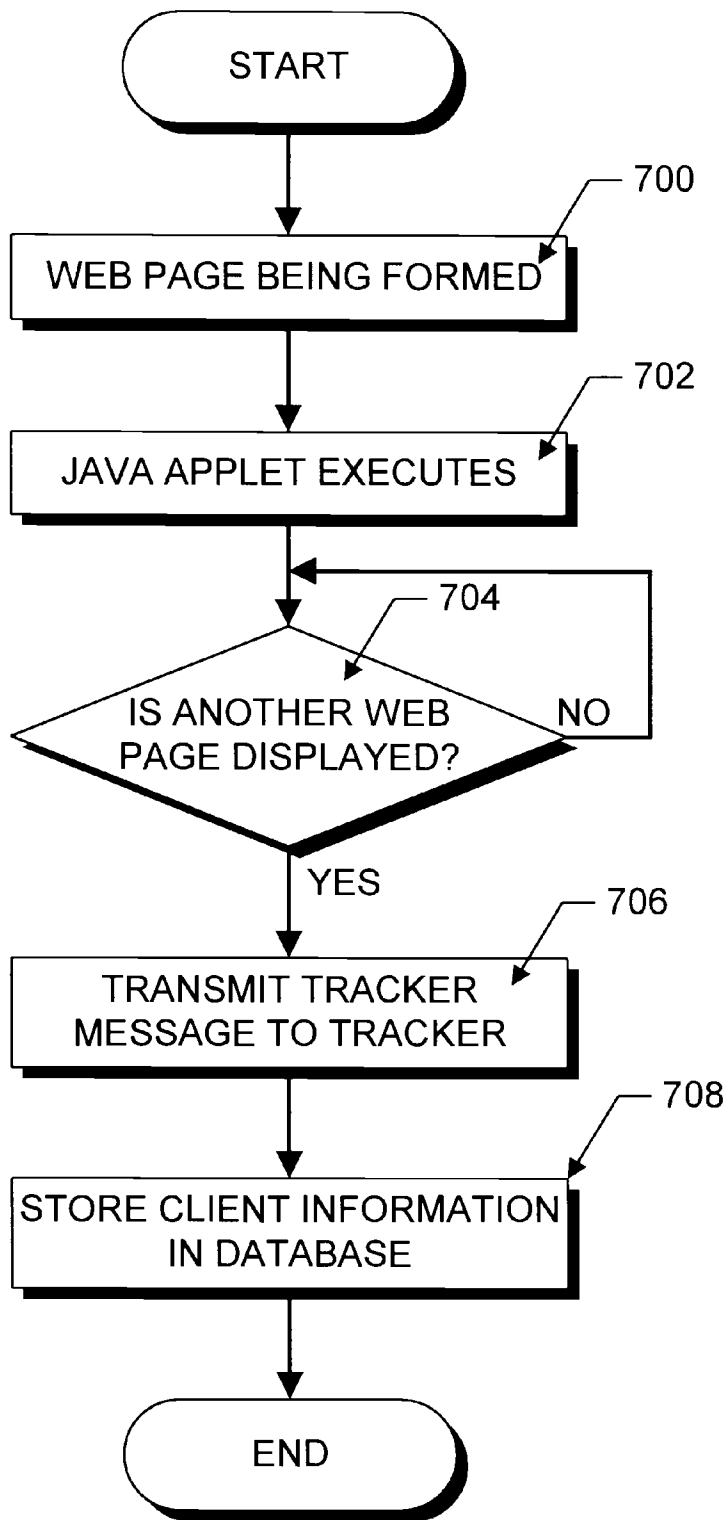


U.S. Patent

Aug. 29, 2000

Sheet 7 of 7

6,112,240



**FIG. 7**

6,112,240

1

## WEB SITE CLIENT INFORMATION TRACKER

### FIELD OF THE INVENTION

This invention relates generally to World Wide Web sites and, more particularly, to a client information tracker for tracking information relating to client use of World Wide Web sites.

### BACKGROUND OF THE INVENTION

FIG. 1 shows a commonly used network arrangement in which a plurality of local computer systems (100) connected to a local area network (LAN 102) access a plurality of remote servers through the Internet 110. Each remote server may include World Wide Web sites (web sites) that each include a plurality of World Wide Web pages (web pages). Each local computer system 100 may access the remote web sites with web browser software, such as Netscape Navigator™, available from Netscape Communications Corporation of Mountain View, Calif.

The World Wide Web is a collection of servers (i.e., web servers) connected to the Internet that utilize the Hypertext Transfer Protocol (HTTP). HTTP is a known application protocol that provides users with access to files (which can be in different formats, such as text, graphics, images, sound, and video) using a standard page description language known as Hypertext Markup Language (HTML). HTML is used to transmit data and instructions between a remote computer (the web server) and a local computer (client computer) in a form that is understandable to the browser software on the client computer.

Information in web pages accessed over the Internet commonly is downloaded into a volatile cache memory in the client computer system. This enables a person visiting a site (visitor) to quickly re-access web page information that was already downloaded, thereby eliminating the need to repeat the relatively slow process of traversing the Internet to access previously viewed web pages. Browsers thus typically include a mechanism for re-accessing the downloaded web pages. The Netscape Navigator™ browser, for example, includes a "BACK" button and a "FORWARD" button on a graphical user interface for such purposes.

In addition to downloading the web page, the web server also may transmit a "cookie" to the browser and receive previously transmitted cookies stored in the permanent memory of the browser. As is well known in the art, a cookie may include information that facilitates access to the downloaded web page on a subsequent access by the browser. By way of example, such information may enable the client computer to more easily form the web page when it is subsequently displayed by a display device in the client computer system.

Information relating to visitor use of a web site is important in maintaining and supporting a web site. For example, use of a web page by a very small number of visitors might suggest that such web page should be eliminated or redesigned to attract more visitors. One known method of obtaining such information utilizes a counter that counts the number of times a web page has been accessed. The counter typically is in the form of an application program on the web server that may be accessed by a web server administrator. One problem with this method, however, is that it does not account for re-accesses to web pages after such web pages have been downloaded into the cache memory in the client computer. More particularly, this method does not count the number of times a visitor re-accesses the downloaded web

2

page, such as by selecting a BACK button or FORWARD button displayed by the web browser. Such method therefore merely indicates the number of visitors to a web page and not the actual number of times such web page has been accessed. These two numbers can vary significantly, thus providing inaccurate information.

Accordingly, it would be desirable to have a method and apparatus that accurately and efficiently obtains and stores information relating to use of a web site.

### SUMMARY OF THE INVENTION

In accordance with one aspect of the invention, a method and apparatus for obtaining client information relating to a web page in a World Wide Web site utilizes a tracker tag in the code of the web page for initiating a client information tracking program. The tracking program may be on a client computer that is accessing the web page, or a tracking computer that is remote from the client computer. The tracking program is initiated by a tracker message transmitted from a web browser on the client computer to the tracking program when the tracker tag is read by the web browser. In one embodiment, the tracking program first obtains the client information, and then stores the client information in the memory of a computer having the tracking program.

In accordance with another aspect of the invention, the apparatus for obtaining client information includes a mechanism for intercepting a request from the browser to display a previously downloaded web page on a client display device at the client computer. The invention further includes a mechanism for controlling the client computer to direct a second tracker message to the tracking program for notifying the tracking program that the web page is displayed on the client display device. The second message therefore notifies the tracker that the web page is being re-accessed by the client computer.

In yet another aspect of the invention, the tracker tag is a Java applet contained within the code of the web page. The applet executes responsively executes when the web page being formed for display, thus obtaining the client information. The client information then is directed to the tracking computer for the tracking program. Each time the downloaded web page is displayed by the client computer, the applet obtains further client information and again directs such information to the tracking computer.

### BRIEF DESCRIPTION OF THE DRAWINGS

The foregoing and other objects and advantages of the invention will be appreciated more fully from the following further description thereof with reference to the accompanying drawings wherein:

FIG. 1 is a block diagram of a commonly used network arrangement.

FIG. 2 is a schematic illustration of a computer architecture on which the invention may be implemented.

FIG. 3 is a block diagram of a network that may be utilized by the disclosed system.

FIG. 4 is a flow chart generally summarizing the steps of obtaining and storing client information.

FIG. 5 is a flow chart summarizing the steps that may be used by a Javascript enabled browser for obtaining client information.

FIG. 6 is a flow chart summarizing the steps that may be used by most currently existing browsers for obtaining client information.

6,112,240

3

FIG. 7 is a flow chart summarizing the steps that may be used by a Java enabled browser for obtaining client information.

#### DETAILED DESCRIPTION OF ILLUSTRATIVE EMBODIMENTS

FIG. 2 illustrates the system architecture for an exemplary computer system 200, such as an IBM THINKPAD 701® computer, that can be utilized to access a World Wide Web page (web page) at a World Wide Web site (web site). Information relating to the client computer 200 and its access and use of the web page (i.e., client information) can be obtained and utilized by a system administrator of the web site. The exemplary computer system of FIG. 2 is discussed only for descriptive purposes, however, and should not be considered a limitation of the invention. Although the description below may refer to terms commonly used in describing particular computer systems, such as an IBM THINKPAD 701® computer, the described concepts apply equally to other computer systems, including systems having architectures that are dissimilar to that shown in FIG. 2.

The computer 200 includes a central processing unit (CPU) 205, which may include a conventional microprocessor, random access memory (RAM) 270 for temporary storage of information, and read only memory (ROM) 215 for permanent 210 storage of information. A bus controller 225 is provided for controlling bus 230, and an interrupt controller 235 is used for receiving and processing various interrupt signals from the other system components.

Mass storage may be provided by diskette 242, CD-ROM 247, or hard disk 252. Data and software may be exchanged with computer 200 via removable media, such as diskette 242 and CD-ROM 247. Diskette 242 is insertable into diskette drive 241, which is connected to bus 230 by controller 240. Similarly, CD-ROM 247 is insertable into CD-ROM drive 246, which is connected to bus 230 by controller 245. Finally, the hard disk 252 is part of a fixed disk drive 251, which is connected to bus 230 by controller 250.

User input to the computer 200 may be provided by a number of devices. For example, a keyboard 256 and a mouse 257 may be connected to bus 230 by keyboard and mouse controller 255. An audio transducer 296, which may act as both a microphone and a speaker, is connected to bus 230 by audio controller 297. It should be obvious to those reasonably skilled in the art that other input devices, such as a pen and/or tablet and a microphone for voice input, may be connected to computer 200 through bus 230 and an appropriate controller. DMA controller 260 is provided for performing direct memory access to system RAM 210. A visual display is generated by a video controller 265, which controls video display 270.

Computer system 200 generally is controlled and coordinated by operating system software, such as the WINDOWS 95® operating system (available from Microsoft Corp., Redmond, Wash.). Among other computer system control functions, the operating system controls allocation of system resources and performs tasks such as process scheduling, memory management, networking and I/O services.

Computer 200 also includes a network adapter 290 that allows the computer 200 to be interconnected to a network 295 via a bus 291. The network 295 may be a local area network (LAN), a wide area network (WAN), or the Internet.

FIG. 3 shows a network 295 that may be utilized by a preferred embodiment of the invention. The network 295

4

includes the client computer 200 having a web browser (browser 302), a web server 304 having a web site 306 that is accessible by the browser 302, and a tracking computer 308 having a tracking program (tracker 310) for obtaining client information relating to the web pages in a web site 306. The client computer 200, web server 304, and tracking computer 308 communicate through the Internet 295A. The browser 302 initiates access to the web site 306 by transmitting a browser request message to the web site 306. The browser request message includes a request to access the site, and a browser header that includes information relating to the browser 302 and client computer 200. This information may include the browser type and version, the type of client computer, and the operating system used by the client computer 200.

In the preferred embodiment of the invention, the process for gathering and storing client information on the tracker 310 is performed each time a web page from the web site 306 is displayed by the client computer 200. FIG. 4 is a flow chart illustrating a generalized process for gathering and storing the client information for a web page in a web site 306. The process begins at step 400 where a tracker tag is added to the HTML code of the web page. The tracker tag, which may be added either manually or by means of a conventional application program, is code written in a preselected form that typically produces a known HTML <IMG> tag within the HTML code of the web page. The preselected form may be any known form, such as Javascript, Java, or conventional HTML.

At step 402, the execution of the tracker tag by the browser 302 creates a tracker message when the web page is being formed for display by the client computer 200. The tracker message may be transmitted to the tracking computer 308 either immediately after the web page is displayed, or at a later time after the page is displayed. The tracker message may include client information (discussed below) and/or may be a call to the tracking computer 308 to execute the tracker 310. At step 404, the tracker 310 obtains the client information in accordance with any method that may facilitate information retrieval. The client information then is stored in a client information database on the tracking computer 308 (step 406). Details of three implementations of the process shown in FIG. 4 are discussed with reference to FIGS. 5-7.

An administrator of the web site 306 may access the client information database at any time to ascertain how the web page has been used. To that end, the tracking computer 308 may include a database retrieval program, such as a Java applet, that enables the client information to be remotely retrieved from the client information database. The administrator may access the Java applet via a Java enabled browser 312 on the web server 304. The retrieved client information then may be displayed on a display device (not shown) on the web server 304. The web page may be modified as needed based on the displayed client information. For example, if most visits to the web page are for an insignificant period of time, the administrator may modify the web page to include more sophisticated graphics, or to include information that is more useful for prospective web page users.

FIG. 5 is a flow chart showing a first implementation of the process shown in FIG. 4. In summary, this first implementation utilizes a tracker tag, written in Javascript (developed by Netscape Communications Corporation), to direct a tracker message from the client computer 200 to the tracker 310 each time the web page is formed for display by the display device on the client computer system 200. The

6,112,240

5

flow chart in FIG. 5 shows the steps for obtaining client information for a web page that is downloaded onto the client computer. The process shown in FIG. 5 may be used by each browser 302 accessing the web site 306.

The first implementation begins at step 500 where a request to display the downloaded web page is detected. This may be done by intercepting browser requests on the client computer 200. The process continues to step 502 if such a browser request is detected. At step 502, the browser 302 reads and executes the tracker tag in the downloaded web page while the web page is being formed on the client display device.

As shown by example below, the tracker tag is configured so that each time it is executed by a Javascript enabled browser 302, it appears as an <IMG> tag referring to a file having a URL with an arbitrary number as an extension. The arbitrary number, for example, may be the number of seconds between Jan. 1, 1990 and the time that the tracker tag is executed by the browser. Accordingly, when the <IMG> tag is read by the browser, it first searches the cache memory of the client computer 200 for a file having the selected URL. This file will not be located in the local cache, however, because the browser 302 can only retrieve files having the exact URL of that requested by the <IMG> tag. The arbitrary number therefore prevents such file from being located.

When the file having the selected URL is not located in the local cache, the client computer 200 transmits a tracker message to the tracker 310 on the tracking computer 308 (step 504). The details of the tracker tag and its interaction with the browser 302 and the HTML code in the web page are discussed below. The received tracker message causes the tracker 310 to execute and record the time of the web page access in a database (step 506).

The process then continues to step 508 where a banner image is uploaded from the tracker 310 to the client computer 200 and added to the information in the downloaded and cached web page. The banner image may be a graphic image file (i.e., represented by the extension "gif") representing any indicia, such as the company logo of the software company that manages the tracking computer 308. Alternatively, the banner image may be a transparent one pixel image. The banner image nevertheless provides the response that the browser 302 should receive when the <IMG> tag is read and executed.

At step 510, it is ascertained if the web browser 302 is still on. If it still is on, the process loops back to step 500. The process ends if the browser 302 is off.

This first implementation records the time of each web page access. The client information database therefore may be a relational database having the fields "total number of accesses" and "time of accesses." Other information may be obtained by conventional means, such as the type of browser 302 accessing the web page, the type of computer accessing the web page, and the time between accesses of the web page.

6

As noted above, the tracker tag preferably includes Javascript code that is embedded in the HTML code of the web page. Below is an exemplary Javascript tracker tag code fragment that may be used for the first implementation.

```
5  <script><!--Hide from old browsers
    date=new Date( );
    document.writeln ("<img src=\"http://
    speedracer.taligent.com/cgi-bin/track?_type=1\"
    -"&_s=" date.getTime( )+"\">");
10 // Stop hiding -->
    </script>
```

To Javascript enabled browsers (e.g., Netscape Navigator™ 2.0, 3.0, or Microsoft Internet Explorer™ 3.0), the above Javascript code appears to the browser 302 as the following HTML code fragment:

```

```

20 where 123456789 is a random number that is unique each time the IMG tag calls the tracker 310. Accordingly, the random number is added to the URL of the file called by the <IMG> tag. As previously noted, the browser 302 cannot locate such file in the cache of the client computer. Consequently, the browser 302 directs a tracker message to the tracker 310 (on the server "speedracer.taligent.com"), which, in turn, causes the tracker 310 to record the time of the web page access.

Non-Javascript enabled browsers read the above tracker tag as:

```
30 
```

This HTML code similarly sends a tracker message to the tracker 310 when the tracker tag is read and executed. The tracker 310 responsively records the web page access. In addition, the tracker 310 returns the banner image with a header having instructions to not store the banner image in the local cache. The header may be as follows:

```
40 Content-type: image/gif
    Content-length: 1234
    Pragma: no-cache
    Cache-control: no-cache
    expires: now
    <returned GIF image of length 1234>
```

FIG. 6 shows a second implementation that may be used for obtaining the client information. When used with this implementation, the tracker 310 uses cookies and common gateway interface (CGI) scripts to obtain the client information. An exemplary tracking tag may be as follows:

```
50 
```

The second implementation begins at step 600 in which the web page is being formed for display by the browser 302 on the client computer 200. The web page may be formed in response to either a direct access, or by the selection of the BACK or FORWARD buttons in the browser 302. At step 602, the browser 302 then reads and executes the tracker tag, which causes a tracker message to be directed from the browser 302 to the tracker 310 on the tracking computer 308. As in the first implementation, the browser 302 cannot locate a file in the cache memory of the client computer 200 having the URL following the <IMG> tag, thereby causing the tracker message to be directed from the browser 302 to the tracker 310. This tracker message invokes the tracker 310, which responsively requests and receives the client information (step 604). The information may be obtained from information in the message header of the tracker



6,112,240

7

message, which includes information relating to the browser **302** and client computer **200**. In addition, the tracker **310** receives the last cookie, if any, that the web page received from the tracker **310**. The cookie may include a unique identification number identifying the client computer **200** and/or the last web page in the web site **306** that was visited by such browser **302**. From this information, a web site administrator may determine the number of accesses by a particular browser **302** on a specified client computer **200**, and the last web page accessed by such browser **302**.

The tracker **310** then transmits a new cookie to the browser **302** (step **606**) having selected information relating to the web browser **302**, thereby replacing the cookie, if any, that was previously transmitted from the browser **302** to the tracker **310**. The tracker **310** also transmits a banner image (e.g., a graphic image file similar to that used by the first implementation) to the browser **302** at step **608** that is not stored in the local cache memory of the client computer **200**. Accordingly, the browser **302** will not locate the banner image in the cache memory the next time the web page is formed by the browser, thereby causing the browser **302** to transmit another tracker message to the tracker **310**.

The process then continues to step **610** where the client information is stored in the client information database. It should be noted that the second implementation repeats each time the web page is displayed by the client computer. In addition, the second implementation may be used by each computer accessing the web site **306**.

FIG. 7 shows a third implementation in which the tracker tag is a Java applet (applet) and thus, is operable with Java-enabled browsers. The third implementation begins at step **700** where the web page having the applet is formed for display by the client computer **200**. This causes the browser **302** to execute the applet on the client computer **200**, thereby gathering the client information. Among other functions, the applet preferably calculates the time that the web page is displayed by the client computer **200**. It then is ascertained at step **704** if another web page is being displayed on the client display device. When another web page is displayed, the applet transmits the client information (e.g., the time that the web page was displayed by the client computer **200**) to the tracker **310** via a tracker message (step **706**). The tracker **310** then stores the client information in the database at step **708**. This process repeats each time the web page is displayed on by the client computer **200**, whether by direct access, or by the selection of the BACK or FORWARD buttons in the browser **302**.

The tracker **310** may be any program that implements the above described processes. In the preferred embodiment, the tracker **310** is a CGI program written in C++.

In an alternative embodiment, the invention may be implemented as a computer program product for use with a computer system. Such implementation may include a series of computer instructions fixed either on a tangible medium, such as a computer readable media (e.g., diskette **242**, CD-ROM **247**, ROM **215**, or fixed disk **252** as shown in FIG. 2) or transmittable to a computer system, via a modem or other interface device, such as communications adapter **290** connected to the network **295** over a medium **291**. Medium **291** may be either a tangible medium (e.g., optical or analog communications lines) or a medium implemented with wireless techniques (e.g., microwave, infrared or other transmission techniques). The series of computer instructions embodies all or part of the functionality previously described herein with respect to the invention. Those skilled in the art should appreciate that such computer instructions can be written in a number of programming

8

languages for use with many computer architectures or operating systems. Furthermore, such instructions may be stored in any memory device, such as semiconductor, magnetic, optical or other memory devices, and may be transmitted using any communications technology, such as optical, infrared, microwave, or other transmission technologies. It is expected that such a computer program product may be distributed as a removable media with accompanying printed or electronic documentation (e.g., shrink wrapped software), preloaded with a computer system (e.g., on system ROM or fixed disk), or distributed from a server or electronic bulletin board over the network **295** (e.g., the Internet or World Wide Web).

Each of the graphical user interfaces discussed above may be constructed by conventional software programming techniques known in the art. It is preferred that the GUIs be constructed by visual builders, such as OCX™ (available from Microsoft Corp.) or Delphi™ (available from Borland Corp.).

Although various exemplary embodiments of the invention have been disclosed, it will be apparent to those skill in the art that various changes and modifications can be made which will achieve some of the advantages of the invention without departing from the true scope of the invention. These and other obvious modifications are intended to be covered by the appended claims.

Having thus described the invention, what we desire to claim and secure by Letters Patent is:

1. A method operable on a client computer having a memory and a browser program running in the memory, the method obtaining client information relating to usage of a World Wide Web site web page consisting of data and code for displaying the data, which data and code are downloaded into the memory from a web server in response to a browser request, the method comprising the steps of:

- A. inserting a tracker tag in the web page code in the web server;
- B. operating a tracker program at a World Wide Web site;
- C. sending, in response to the detection of the tracker tag in web page code being read by the browser program, a tracker message from the client computer to the tracker program; and
- D. causing the tracker program to obtain the client information in response to receipt of the tracker message.

2. The method as defined by claim 1 further comprising the step of:

- E. storing the obtained client information in a database located at the tracker program World Wide Web site.

3. The method as defined by claim 1 wherein the tracker message includes client information and step D comprises the step of:

- D1. copying the client information from the tracker message into a database.

4. The method as defined by claim 1 wherein step D comprises the steps of:

- D2. directing, responsive to the tracker message, a cookie having client information from the client computer to the tracker; and
- D3. copying the client information from the cookie into a database.

5. The method as defined by claim 1 further comprising the steps of:

- G. intercepting a request generated by the browser to display a web page downloaded into the client computer memory;
- H. controlling the client computer, responsive to the intercepted request, to direct a second tracker message to the tracker.

6,112,240

9

6. The method as defined by claim 1 wherein the client information includes the number of times the web page has been accessed by the client computer.

7. The method as defined by claim 1 further comprising the step of:

I. directing a cookie identifying the client computer to the browser.

8. The method as defined by claim 1 wherein the web server includes a browser, the tracker program being located on a tracking computer that includes a Java applet and a database for storing the client information, the method further comprising the step of:

J. controlling the Java applet, via the browser on the web server, to access the database on the tracking computer.

9. The method as defined by claim 1 wherein the tracker tag is a Java applet.

10. The method as defined by claim 9 wherein step C comprises the step of:

C1. executing the Java applet with the browser when the Java applet is downloaded, to generate the tracker message, the tracker message including the client information.

11. The method as defined by claim 10 wherein step D comprises the step of:

D4. receiving the tracker message; and

D5. storing the client information in the tracker message in a database.

12. An apparatus operable on a client computer having a memory and a browser program running in the memory, the apparatus obtaining client information relating to usage of a World Wide Web site web page consisting of data and code for displaying the data, which data and code are downloaded into the memory from a web server in response to a browser request, the apparatus comprising:

a tracker program at a World Wide Web site;

a tracker tag in the web page code in the web server, the tracker tag including means for sending, in response to detection of the tracker tag in the web page code when the web page code is read by the browser program, a tracker message from the client computer to the tracker program; and

means for causing the tracker to obtain the client information in response to the receipt of the tracker message.

13. The apparatus as defined by claim 12 wherein the sending means is responsive to the tracker tag being downloaded into the memory of the client computer.

14. The apparatus as defined by claim 12 further comprising:

means for storing the obtained client information in a database at the tracker program World Wide Web site.

15. The apparatus as defined by claim 12 wherein the tracker message includes client information, the means for causing comprising:

means for copying the client information from the tracker message into a database.

16. The apparatus as defined by claim 12 wherein the means for causing comprises:

means for directing, responsive to the tracker message, a cookie having client information from the client computer to the tracker; and

means for copying the client information from the cookie into a database.

10

17. The apparatus as defined by claim 12 further comprising:

means for intercepting a request generated by the browser to display the web page;

means for controlling the client computer, responsive to the intercepted request, to direct a second tracker message to the tracker.

18. The apparatus as defined by claim 12 wherein the client information includes the number of times the web page has been accessed by the client computer.

19. The apparatus as defined by claim 12 further comprising:

means for directing a cookie identifying the client computer to the browser.

20. The apparatus as defined by claim 12 wherein the web server includes a browser, the tracker program being located on a tracking computer that includes a Java applet and a database for storing the client information, the apparatus further comprising:

means for controlling the Java applet, via the browser on the web server, to access the database on the tracking computer.

21. The apparatus as defined by claim 12 wherein the tracker tag is a Java applet.

22. The apparatus as defined by claim 21 wherein the sending means comprises:

means for executing the Java applet with the browser when the Java applet is downloaded, to generate the tracker message, the tracker message including the client information.

23. The apparatus as defined by claim 22 wherein the causing means comprises:

means for receiving the tracker message; and

means for storing the client information in the tracker message in a database.

24. A computer program product comprising:

a computer usable medium having computer readable program code thereon operable on a client computer having a memory and a browser program running in the memory, the computer program product obtaining client information relating to usage of a World Wide Web site web page consisting of data and code for displaying the data, which data and code are downloaded into the memory from a web server in response to a browser request, the computer readable program code comprising:

tracker tag program code in the code of the web page;

program code for operating a tracker program at a World Wide Web site;

program code for sending, in response to the detection of the tracker tag program code in web page code read by the browser program, a tracker message from the client computer to the tracker program; and

program code for causing the tracker program to obtain the client information in response to receipt of the tracker message.

25. The computer program product as defined by claim 24 further comprising:

program code for storing the obtained client information in a database.

26. The computer program product as defined by claim 24 wherein the tracker message includes client information, the program code for causing comprising:

program code for copying the client information from the tracker message into a database.

6,112,240

11

27. The computer program product as defined by claim 24 wherein the program code for causing comprises:

- program code for directing, responsive to the tracker message, a cookie having client information from the client computer to the tracker; and
- program code for copying the client information from the cookie into a database.

28. The computer program product as defined by claim 24 further comprising:

- program code for intercepting a request generated by the browser to display a web page downloaded in the memory of the client computer;
- program code for controlling the client computer, responsive to the intercepted request, to direct a second tracker message to the tracker.

29. The computer program product as defined by claim 24 wherein the client information includes the number of times the web page has been accessed by the client computer.

30. The computer program product as defined by claim 24 further comprising:

- program code for directing a cookie identifying the client computer to the browser.

31. The computer program product as defined by claim 24 wherein the web server includes a browser, the tracker program being located on a tracking computer that includes a Java applet and a database for storing the client information, the computer program product further comprising:

- program code for controlling the Java applet, via the browser on the web server, to access the database on the tracking computer.

32. The computer program product as defined by claim 24 wherein the tracker tag is a Java applet.

33. The computer program product as defined by claim 32 wherein the program code for sending comprises:

- program code for executing the Java applet when the Java applet is downloaded, to generate the tracker message, the tracker message including the client information.

34. The computer program product as defined by claim 33 wherein the program code for causing comprises:

- program code for receiving the tracker message; and
- program code for storing the client information in the tracker message in a database.

35. A method operable on a client computer having a memory and a browser program running in the memory, the method obtaining client information relating to usage of a web page consisting of data and code for displaying the data, which data and code are downloaded into the memory from a web server in response to a browser request, the code having tags, each of which includes the address of a tracking server and a location of a resource at the tracking server which is to be retrieved by the browser program, the method comprising the steps of:

- A. inserting a tracker tag in the web page code in the web server, wherein the tracker tag includes a mechanism for modifying the resource location in the tracker tag so that the modified location will be different each time the tracker tag is detected by the browser program;
- B. operating a tracker program at the tracking server;
- C. sending, in response to each detection of the tracker tag in web page code being read by the browser program, a tracker message from the client computer to the tracker program to retrieve a resource at the modified location; and
- D. causing the tracker program to obtain the client information in response to receipt of the tracker message.

12

36. An apparatus operable on a client computer having a memory and a browser program running in the memory, the apparatus obtaining client information relating to usage of a web page consisting of data and code for displaying the data, which data and code are downloaded into the memory from a web server in response to a browser request, the code having tags, each of which includes the address of a tracking server and a location of a resource at the tracking server which is to be retrieved by the browser program, the apparatus comprising:

- a tracker program at the tracking server;

a tracker tag in the web page code in the web server, the tracker tag including a mechanism for modifying the resource location in the tracker tag so that the modified location will be different each time the tracker tag is detected by the browser program and means for sending, in response to each detection of the tracker tag in the web page code when the web page code is read by the browser program, a tracker message from the client computer to the tracker program to retrieve a resource at the modified location; and

means for causing the tracker to obtain the client information in response to the receipt of the tracker message.

37. A computer program product comprising:

a computer usable medium having computer readable program code thereon operable on a client computer having a memory and a browser program running in the memory, the computer program product obtaining client information relating to usage of a web page consisting of data and code for displaying the data, which data and code are downloaded into the memory from a web server in response to a browser request, the code having tags, each of which includes the address of a tracking server and a location of a resource at the tracking server which is to be retrieved by the browser program, the computer readable program code comprising:

tracker tag program code in the code of the web page, wherein the tracker tag program code includes program code for modifying the resource location in the tracker tag so that the modified location will be different each time the tracker tag is detected by the browser program;

program code for operating a tracker program at the tracking server;

program code for sending, in response to the detection of the tracker tag program code in web page code read by the browser program, a tracker message from the client computer to the tracker program to retrieve a resource at the modified location; and

program code for causing the tracker program to obtain the client information in response to receipt of the tracker message.

38. A computer data signal embodied in a carrier wave for obtaining client information relating to usage of a World Wide Web site web page consisting of data and code for displaying the data, which data and code are downloaded into the memory from a web server in response to a browser request, the computer data signal comprising:

program code for inserting a tracker tag into the code of the web page;

program code for installing a tracker program at a tracking server, the tracker program obtaining the client information in response to receipt of a tracker message; and



6,112,240

13

program code for sending, in response to the detection of the tracker tag program code in web page code when read by the browser program, a tracker message from the client computer to the tracker program so that the tracker program monitors the web page usage.

39. The computer data signal as defined by claim 38 further comprising program code for storing the obtained client information in a database.

40. The computer data signal as defined by claim 38 wherein the tracker message includes client information, and wherein the tracker program comprises program code for copying the client information from the tracker message into a database.

41. The computer data signal as defined by claim 38 wherein the tracker program code comprises:

program code for directing, responsive to the tracker message, a cookie having client information from the client computer to the tracker program; and

program code for copying the client information from the cookie into a database.

42. The computer data signal as defined by claim 38 further comprising:

program code for intercepting a request generated by the browser to display a web page downloaded in the memory of the client computer; and

program code for controlling the client computer, responsive to the intercepted request, to direct a second tracker message to the tracker program.

14

43. The computer data signal as defined by claim 38 wherein the client information includes the number of times the web page has been accessed by the client computer.

44. The computer data signal as defined by claim 38 wherein the program code for sending a tracker message comprises program code for directing a cookie identifying the client computer to the tracker program.

45. The computer data signal as defined by claim 38 wherein the web server includes a browser and the tracker program includes a Java applet and a database for storing the client information, the computer data signal further comprising program code for controlling the Java applet, via the browser on the web server, to access the database on the tracking server.

46. The computer data signal as defined by claim 38 wherein the tracker tag is a Java applet.

47. The computer data signal as defined by claim 46 wherein the program code for sending a tracker message comprises program code for executing the Java applet when the Java applet is downloaded, to generate the tracker message, the tracker message including the client information.

48. The computer data signal as defined by claim 47 wherein the tracker program code comprises:

program code for receiving the tracker message; and

program code for storing the client information in the tracker message in a database.

\* \* \* \* \*



US006128655A

**United States Patent** [19]  
**Fields et al.**

[11] **Patent Number:** **6,128,655**  
[45] **Date of Patent:** **Oct. 3, 2000**

[54] **DISTRIBUTION MECHANISM FOR FILTERING, FORMATTING AND REUSE OF WEB BASED CONTENT**

[75] Inventors: **Duane Kimbell Fields**, Austin; **Sebastian Hassinger**, Blanco; **William W. Hurley, II**, Round Rock, all of Tex.

[73] Assignee: **International Business Machines Corporation**, Armonk, N.Y.

[21] Appl. No.: **09/113,678**

[22] Filed: **Jul. 10, 1998**

[51] **Int. Cl.**<sup>7</sup> ..... **G06F 13/00**

[52] **U.S. Cl.** ..... **709/219; 709/225; 707/501**

[58] **Field of Search** ..... **709/202, 203, 709/205, 217, 219, 223, 225, 227, 310; 707/10, 501**

[56] **References Cited**

**U.S. PATENT DOCUMENTS**

4,586,158	4/1986	Brandle .	
5,196,838	3/1993	Meier et al. .	
5,598,188	1/1997	Gove et al. .	
5,602,997	2/1997	Carpenter et al. .	
5,673,322	9/1997	Pepe et al. .	
5,704,017	12/1997	Heckerman et al. .	
5,706,502	1/1998	Foley et al. .	
5,706,507	1/1998	Schloss .	
5,708,780	1/1998	Levergood et al. .	
5,855,020	12/1998	Kirsch	707/10
5,918,010	6/1999	Appleman et al.	709/203
5,918,013	6/1999	Mighdoll et al.	709/217
5,987,606	11/1999	Cirasole et al.	713/200
5,991,760	11/1999	Gauvin et al.	707/10
6,009,429	12/1999	Greer et al.	707/10

**FOREIGN PATENT DOCUMENTS**

WO9726729 12/1996 European Pat. Off. .

WO9727553 1/1997 European Pat. Off. .

**OTHER PUBLICATIONS**

Digestor: Device-Independent Access To The World Wide Web (<http://www.fxpal.xerox.com/papers/bic97>).

IBM Technical Disclosure Bulletin, vol. 40 No. 12 p. 143, Dec. 1997—Look Ahead Filtering of Internet Content.

IBM Technical Disclosure Bulletin, vol. 40 No. 12 p. 181, Dec. 1997—Filtering Internet Content.

IBM Technical Disclosure Bulletin, vol. 40 No. 12 pp.5–8, Dec. 1997—Method for Dynamically Routing Web Requests to Different Web Servers.

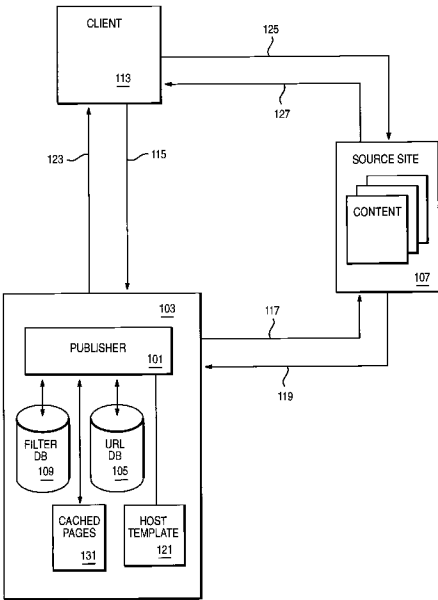
IBM Technical Disclosure Bulletin, vol. 40 No. 07 pp. 1–4, Jul. 1997—Service to Enable Common Gateway Interface Programs within Tivoli Management Environment Netfinity Based Internet Applications.

*Primary Examiner*—Viet D. Vu  
*Attorney, Agent, or Firm*—Jeffrey S. LaBaw

[57] **ABSTRACT**

The invention provides an automated system for replicating published web content and associated advertisements in the context of a hosting web site. At the hosting web site, the invention includes the process of brokering a client browser's request for a web page, analyzing the returned content and splitting it into component elements, extracting the desired component elements, recasting the desired elements in the look and feel of the hosting site and sending the recast content to the requesting client as a web page. Once the reformatted file is received at the client, the client browser interprets the HTML in the web page, presenting the content in the context of the hosting web site. On the content provider's web site, the details of the transaction in the web server logs are preserved, proxying a direct page view and ad impression.

**37 Claims, 11 Drawing Sheets**



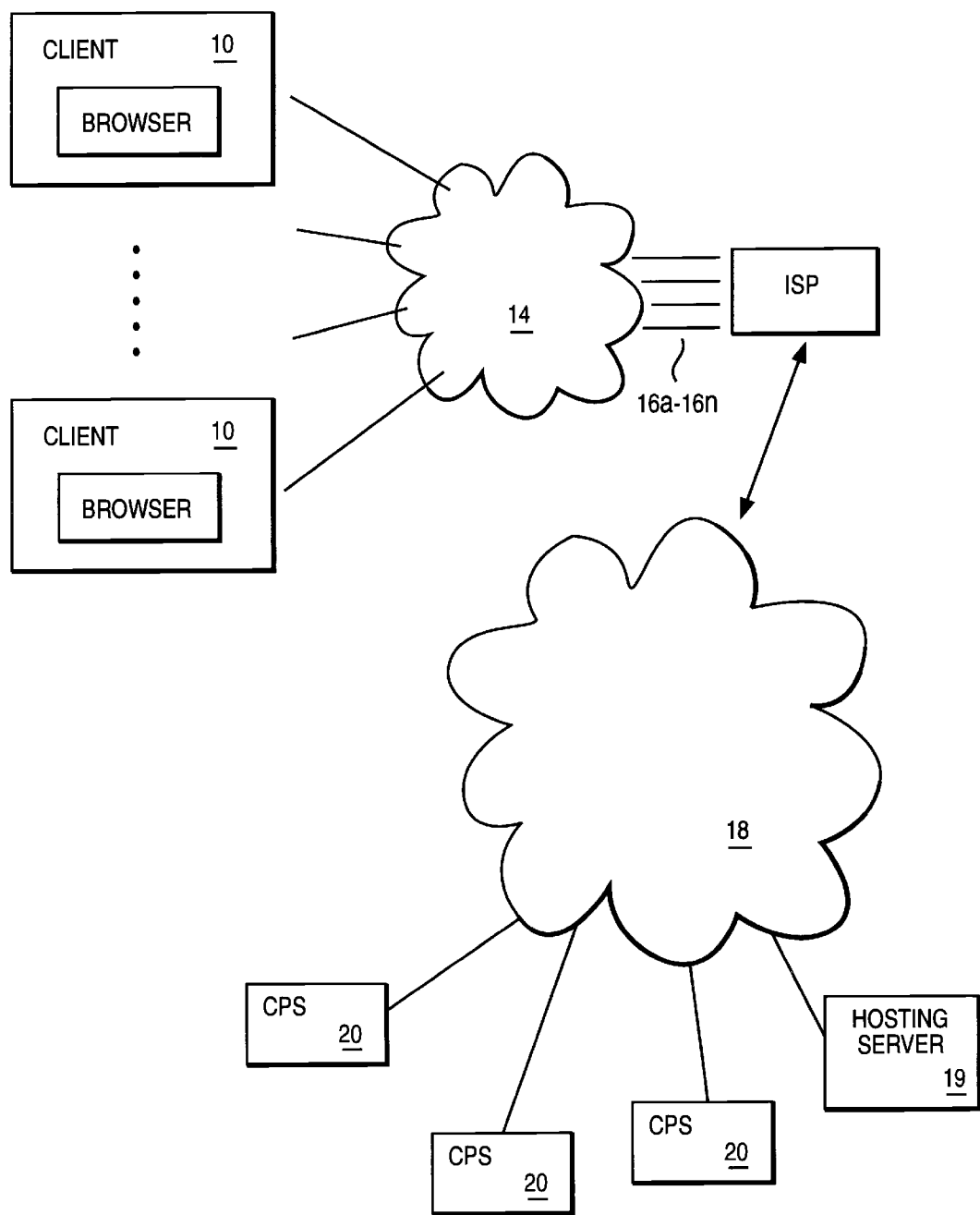


FIG. 1

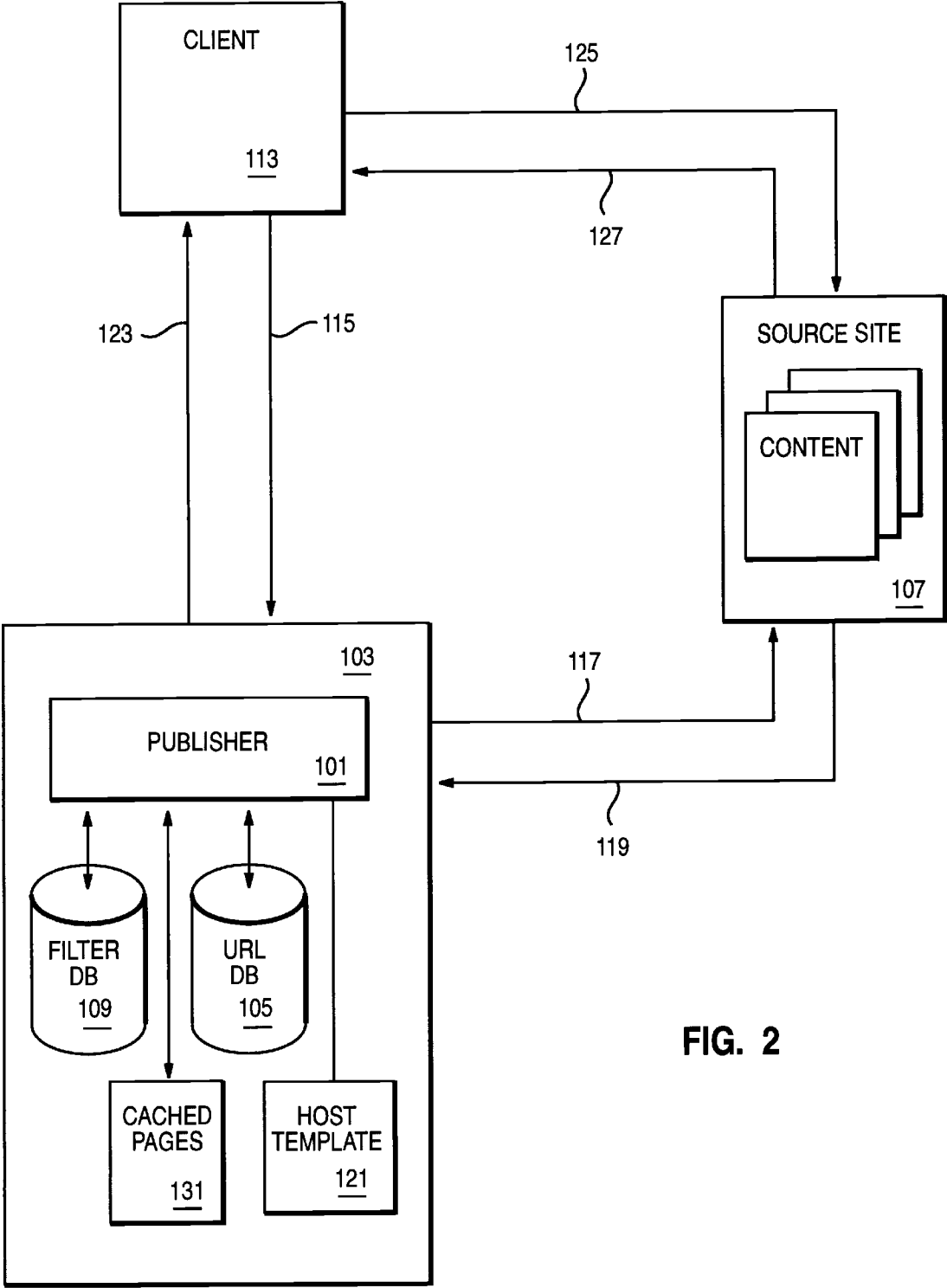


FIG. 2

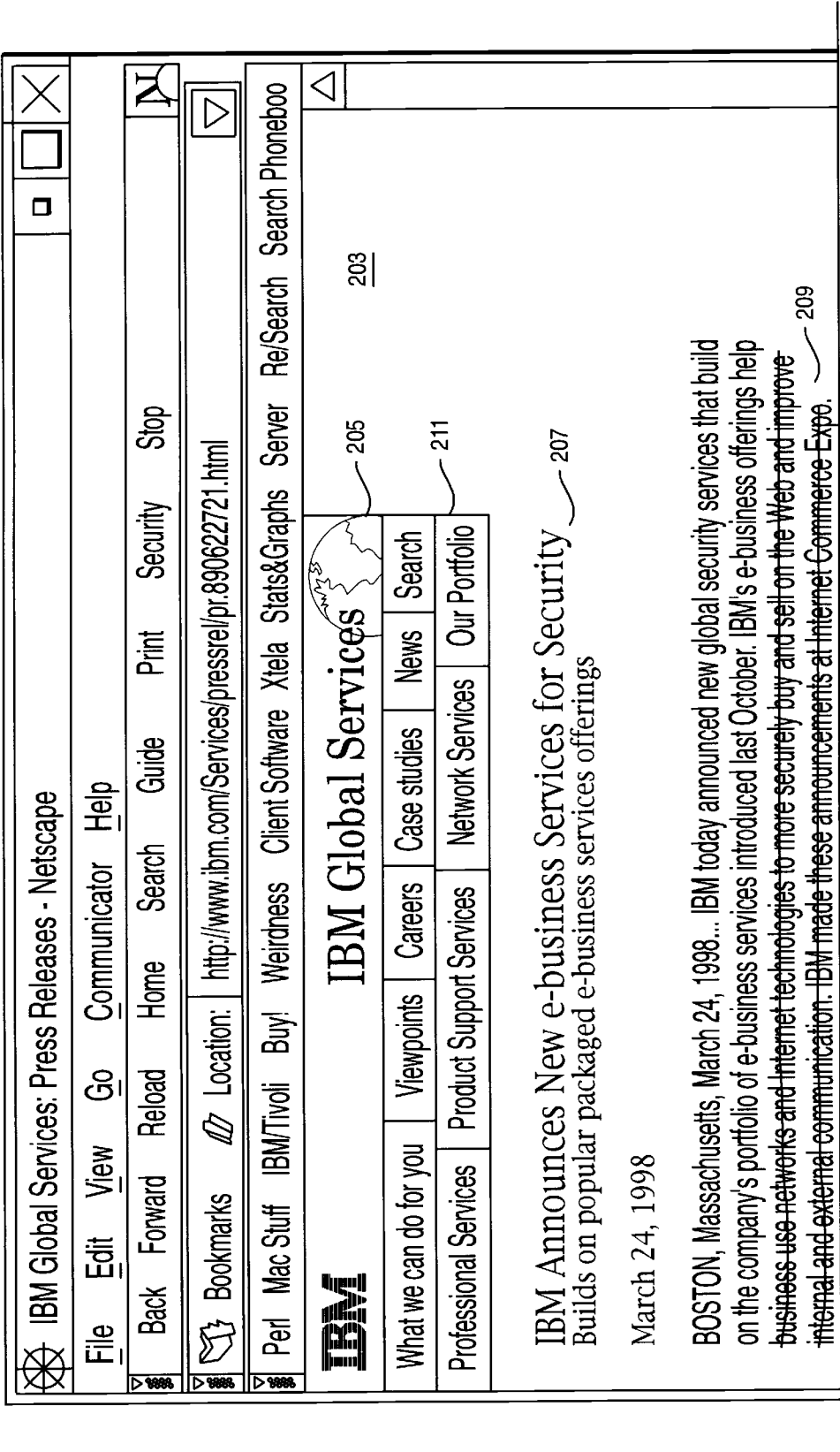


FIG. 3A

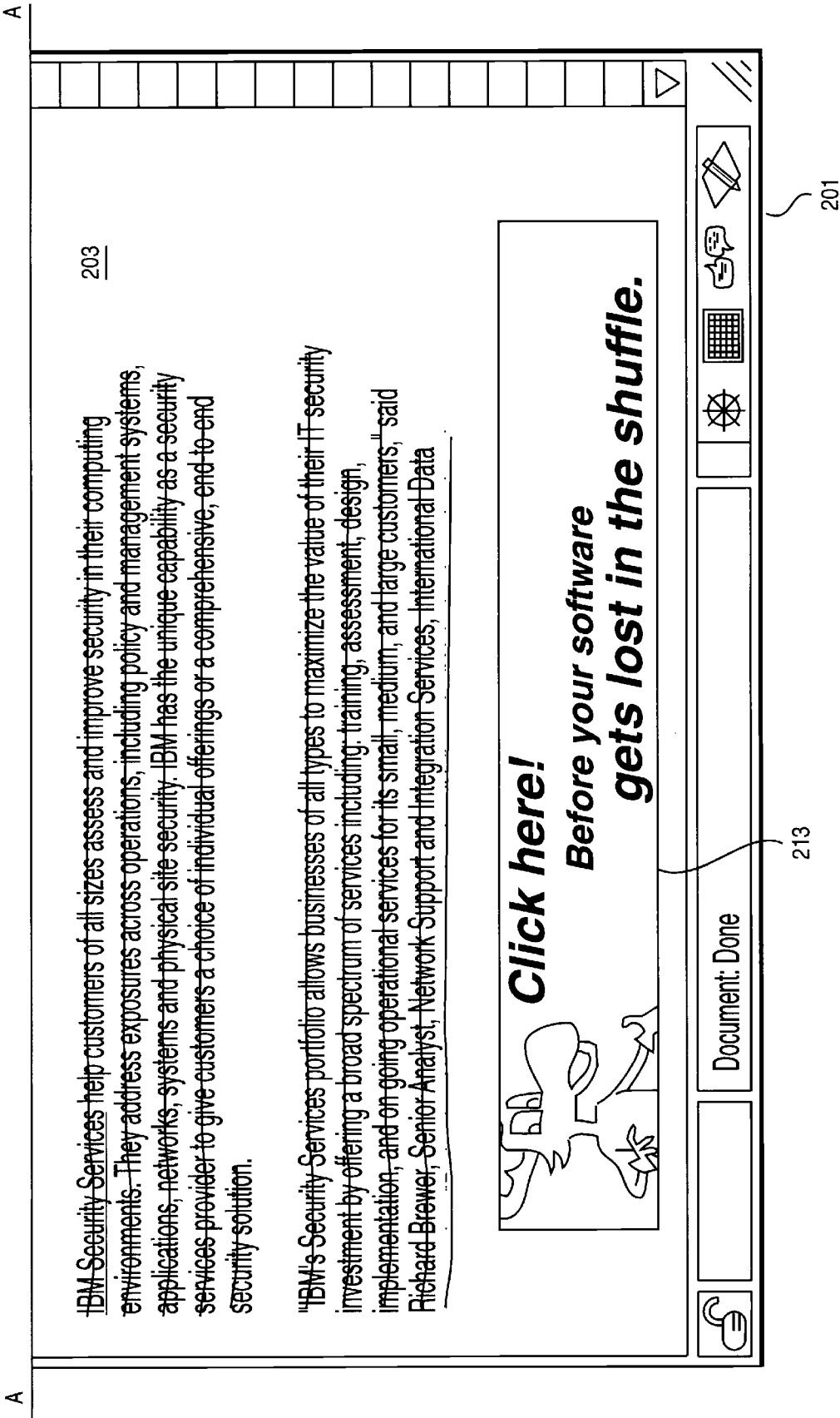


FIG. 3B

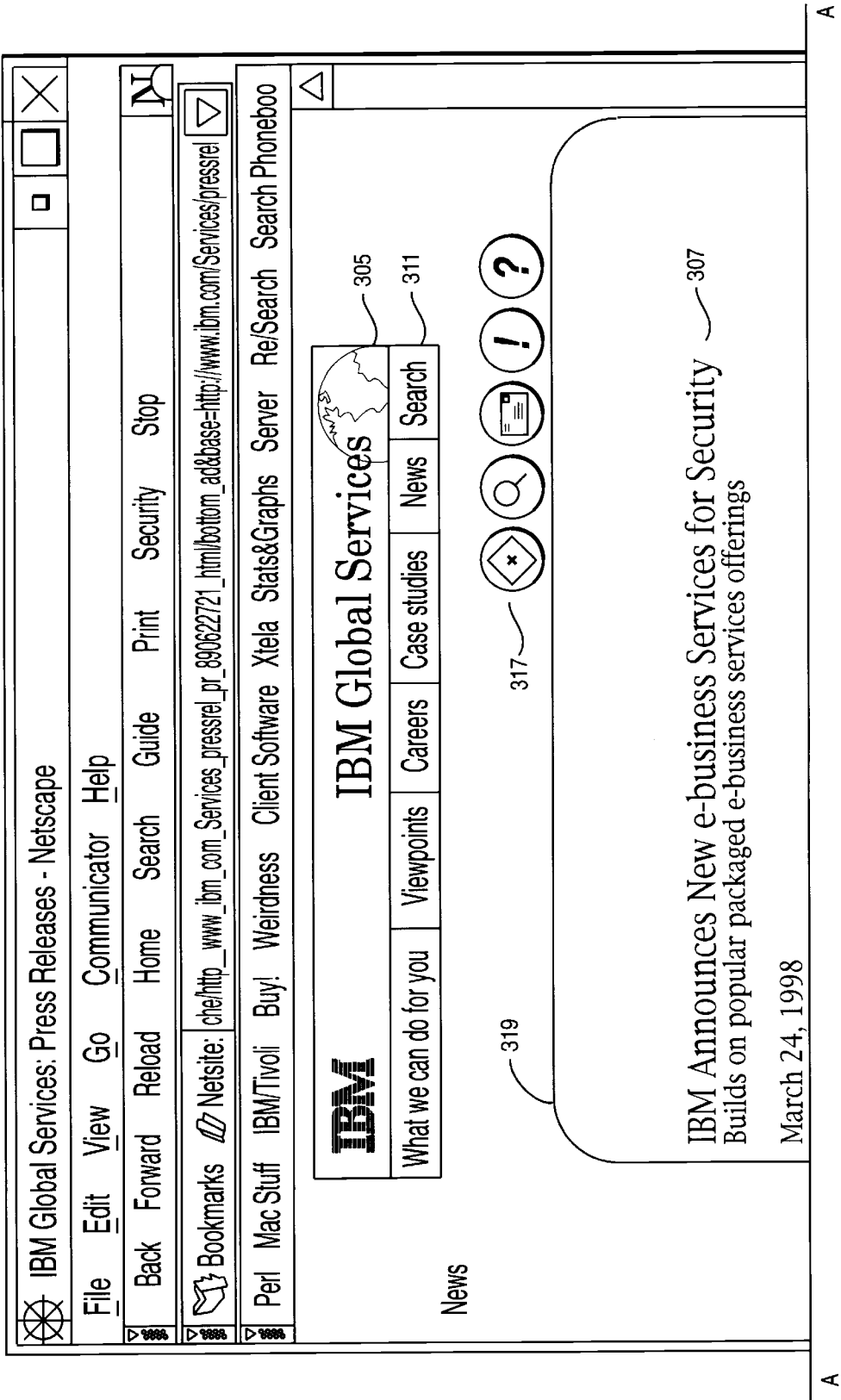


FIG. 4A



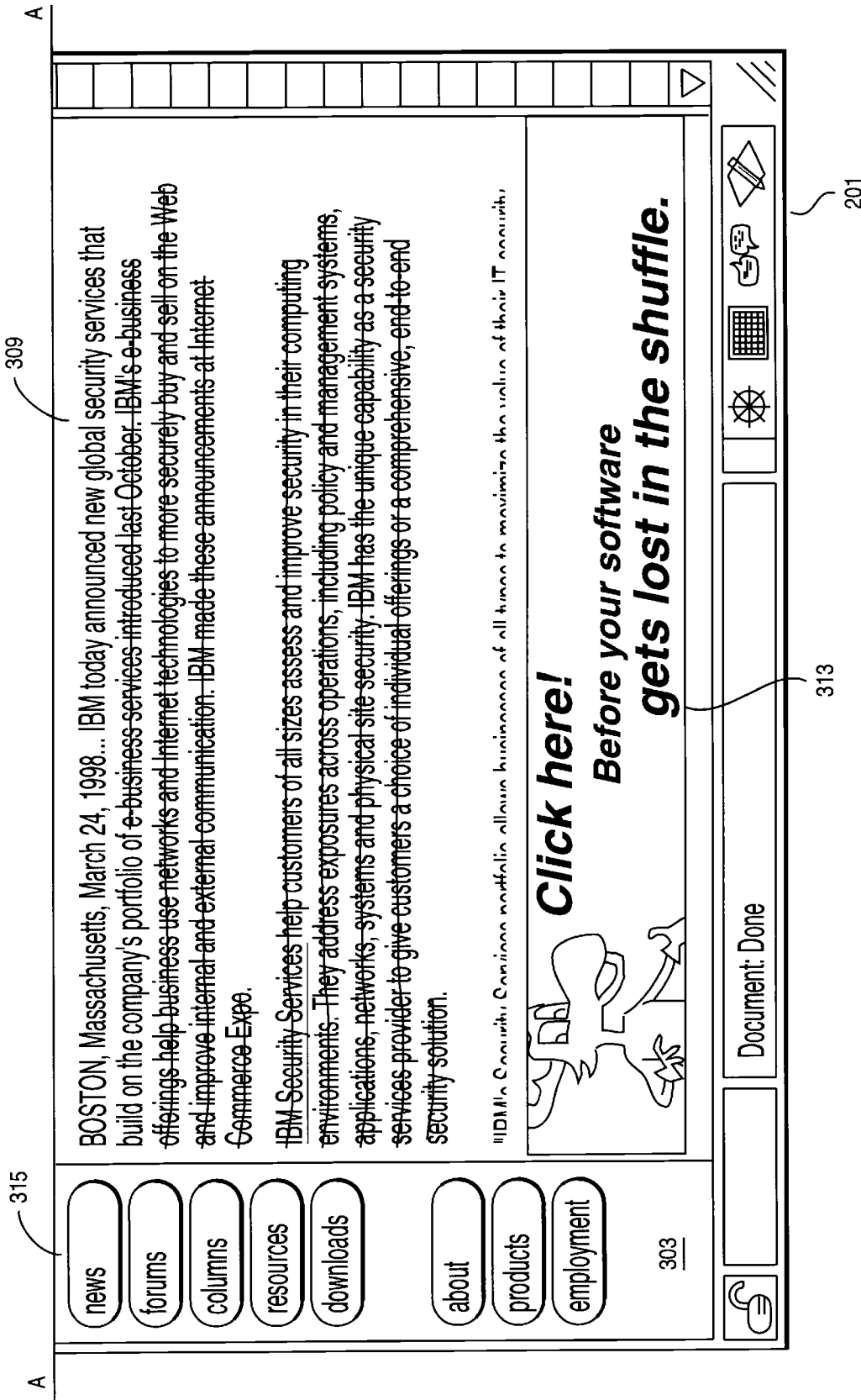
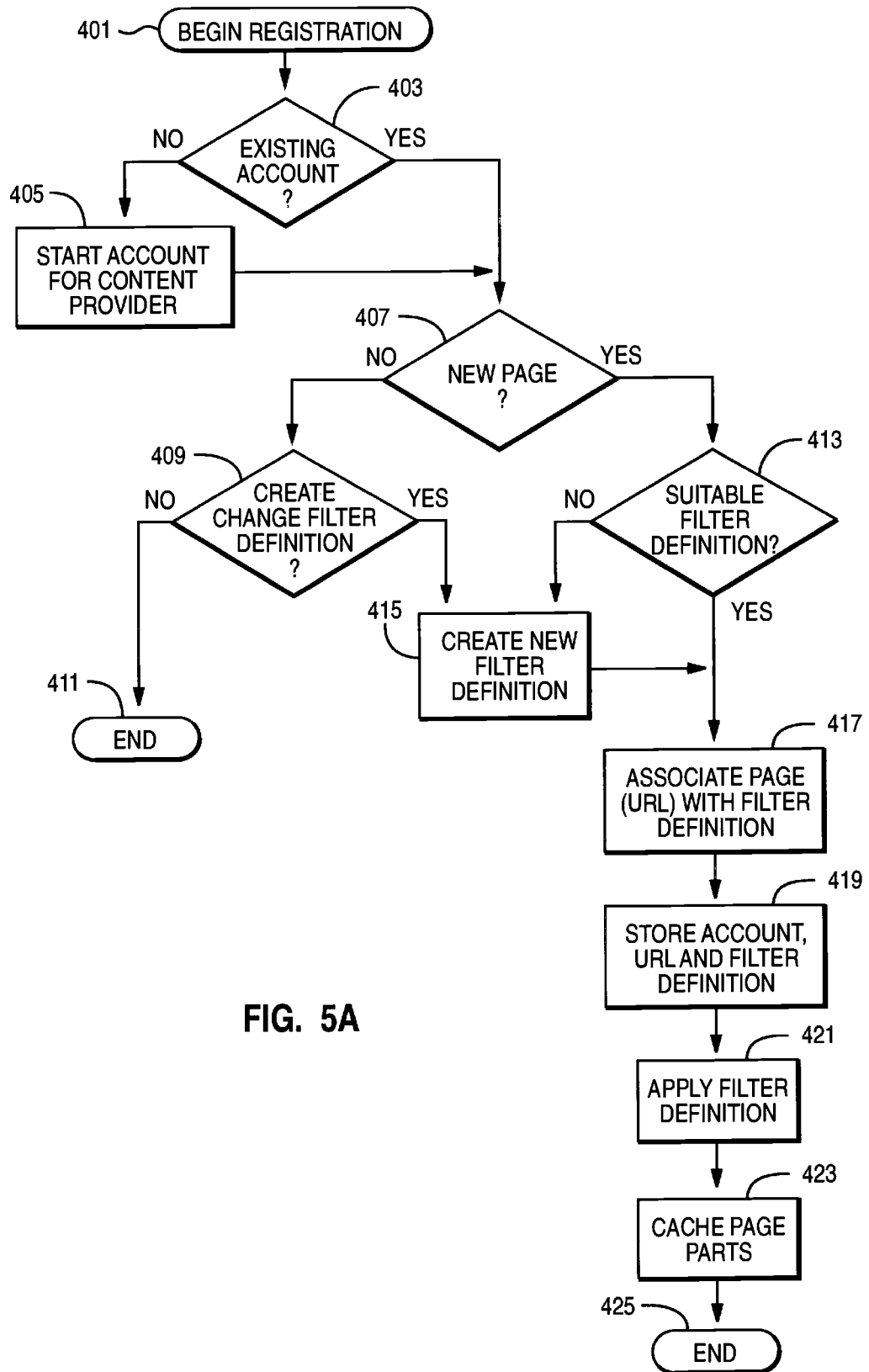


FIG. 4B



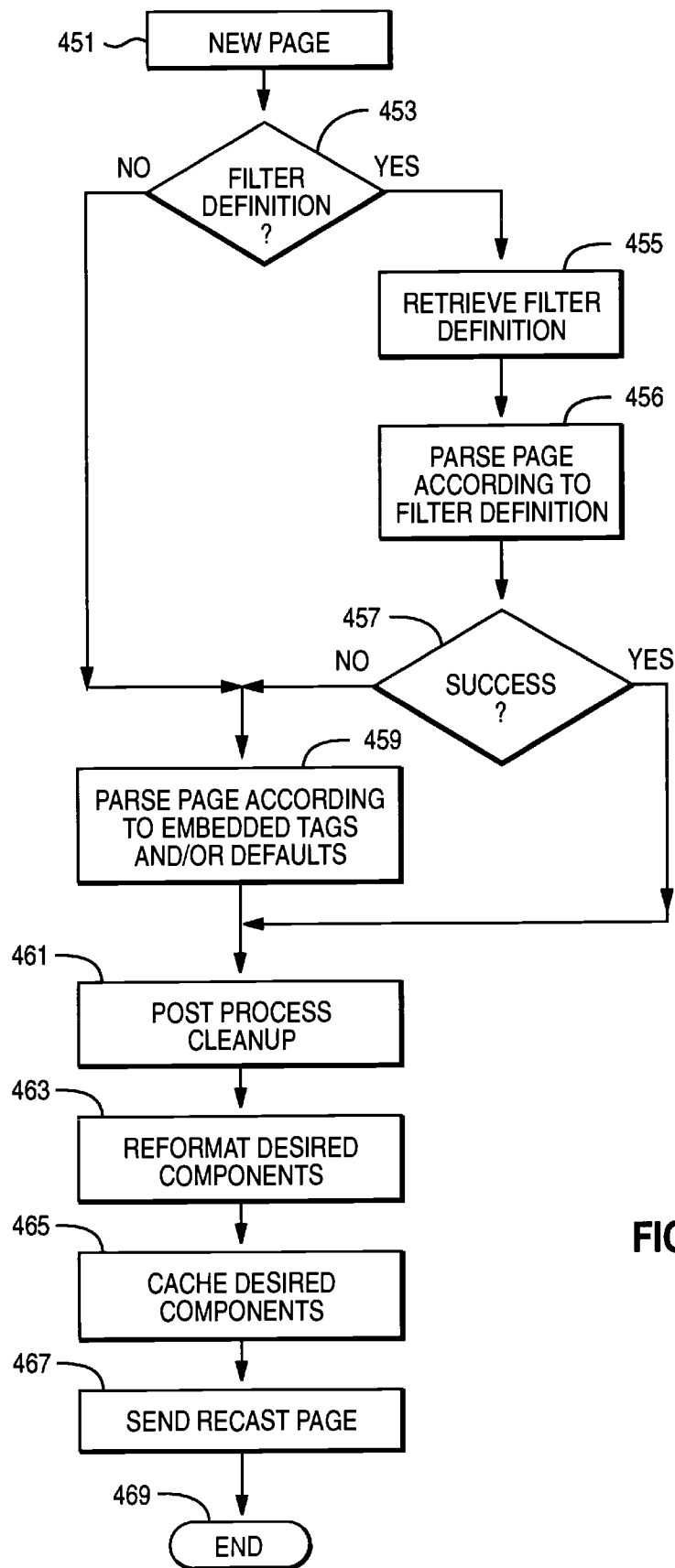


FIG. 5B

### Pass-through filter editing interface showing a typical filter:

The screenshot shows the Netscape Communicator interface. The title bar is 'Netscape Communicator Help'. The menu bar includes 'File', 'Edit', 'View', 'Go', 'Communicator', and 'Help'. The toolbar contains icons for 'Back', 'Forward', 'Reload', 'Home', 'Search', 'Guide', 'Print', 'Security', and 'Stop'. The Bookmarks toolbar shows a 'Netsite' icon and the URL 'http://www.cross-site.com/cgi-bin/make\_filter.pl'. The main content area displays a Perl script for a web page, including a filter name, logo name, copyright string, and several banner and article positions with associated line numbers (505, 507, 509, 511, 513, 515, 517, 519).

A

Ending of bottom banner ad:  

521

 keep: ☐

Strip table formatting from:  
☒ Article: ☐ Bottom ad area: ☐ 525

Custom filter code:  
{  
    my (&top\_banner, &article, &bottom\_banner) = @ \_; 527  
}

529

503

return (&top\_banner, &article, &bottom\_banner);

531

Change Filter

Document: Done

501

A

FIG. 6B

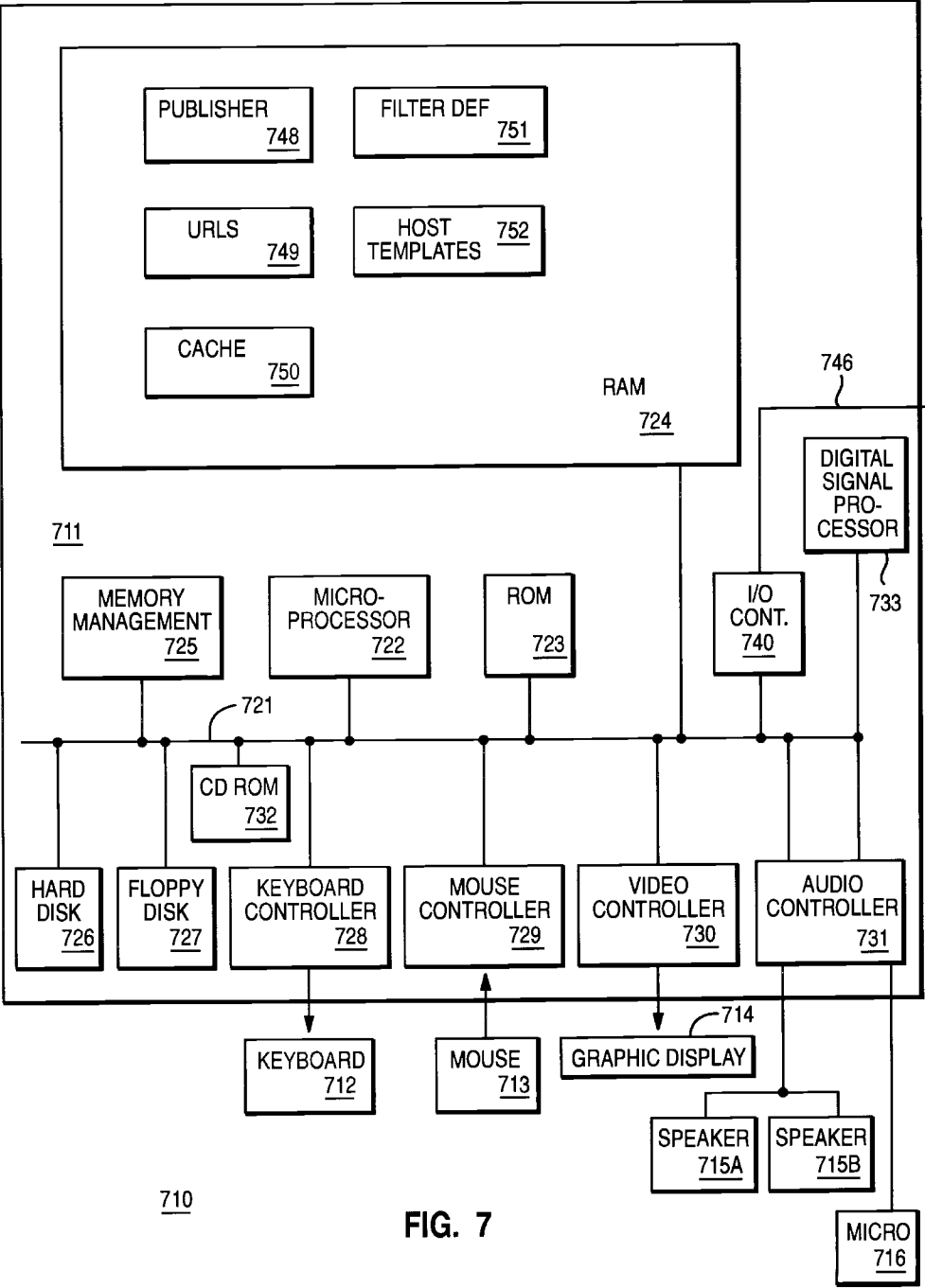


FIG. 7

6,128,655

1

## DISTRIBUTION MECHANISM FOR FILTERING, FORMATTING AND REUSE OF WEB BASED CONTENT

### BACKGROUND OF THE INVENTION

The present invention relates generally to the data processing systems. More particularly, it relates to managing and formatting electronically-published material distributed over a computer network.

The World Wide Web is the Internet's multimedia information retrieval system. In the Web environment, client machines effect transactions to Web servers using the Hypertext Transfer Protocol (HTTP), which is a known application protocol providing users access to files (e.g., text, graphics, images, sound, video, etc.) using a standard page description language known as Hypertext Markup Language (HTML). HTML provides basic document formatting and allows the developer to specify "links" to other servers and files. In the Internet paradigm, a network path to a server is identified by a so-called Uniform Resource Locator (URL) having a special syntax for defining a network connection. Use of an HTML-compatible browser (e.g., Netscape Navigator or Microsoft Internet Explorer) at a client machine involves specification of a link via the URL. In response, the client makes a request to the server (sometimes referred to as a "Web site") identified in the link and, in return, receives in return a document or other object formatted according to HTML.

Among the many challenges in running a successful web site is the constant creation and updating the web pages and other files, i.e. web content, to keep the site fresh and new and attractive to web users. Web sites which do not update their content on a regular basis tend to lose their favor. Eventually, fewer "hits" are logged on the web site's pages as fewer users view the information or advertisements which the web site is publishing. As web based advertising fees are typically based on the number of hits a page or site receives, this reduction will directly and adversely affect the revenues of the web site. Of course, the constant update of the web content, while necessary to maintain the popularity of the site, is very expensive in terms of manpower and time.

Furthermore, much of the information on a particular web site is redundant when compared to information available on other similar sites. Some of this duplicate information represents differences in opinion and is no doubt the sign of a tolerant and free society. However, much of the information is simply a duplication of the same news on each web site. From the perspective of the web site content provider, it would be efficient if some of the information found on other sites could be reused or "hosted" on his site. Thus, additional manpower for writing and entering articles on the web server can be reduced or eliminated. Of course, such reuse is subject to the copyright laws and must be the subject of an agreement with the content provider of the source material.

While Web-based content exists in abundance, it is not necessarily easy to persuade a web content provider to share content on a low or no charge basis. This is especially true for Web-based news articles, as these news articles typically represent the major revenue generating content for the publisher by carrying advertising banners above and/or below the article text. Therefore, the web publishers are apt to charge a large amount for licensing the content to other sites for reprinting. Each reprint represents a loss of revenue under the standard arrangement of exporting the content in raw format to the licensing host and that host posting the articles on their own site without the publisher's advertisements.

2

Further, even if a web site operator could find a content provider willing to share their content at economically favorable terms, other problems exist. A single content provider may not be likely to provide the complete gamut of articles which the hosting web site would like to serve to its web clients. It would be preferable that the hosting site be able to use content from a variety of potential content providing web sites. Again, the likelihood of finding many willing quality web content providers is even lower. Yet even if this feat were accomplished, as each site has its own look and feel, if the content was presented in the format as it originally appeared on each of the web sites, the hosting site would present a disjointed hodgepodge collection of material. It is hardly the professional image that the hosting site should ideally project.

It is unlikely that a web content provider who is essentially sharing his content for free will be willing to install special software or specially format his information for the hosting site. If the material comes in raw format, considerable manpower must thus be devoted to making borrowed material on the hosting site look as though it was specifically created for the site. This effort is naturally compounded where material comes from a range of web content providers. Further, there is likely to be some lag between the time that the web content is available on the content provider's web page and its appearance on the hosting site. This dilutes the desired appearance of the hosting site having the latest and greatest material.

In reality, the hosting site is unlikely to find many partners without some convincing demonstration that its reuse of the material will somehow benefit the original content provider in some way, much less endanger his revenue stream.

The present invention solves this important problem.

### SUMMARY OF THE INVENTION

It is an object of the invention to reduce the expense and effort of providing content in a new hosting web site.

It is another object of the invention reuse content from other web sites with little to no licensing fees.

It is another object of the invention to allow a content provider web site to maintain or expand a revenue base through ad impression.

It is another object of the invention to reuse content from a variety of different content providers some of which may use radically different formats and other content.

It is another object of the invention to adapt content from other web sites to the appearance of the hosting web site so that the content from a plurality of web sites appears native to the hosting web site.

It is another object of the invention to automatically update material on the hosting web site as it changes on the content provider web sites.

It is another object of the invention to reuse web content in a plurality of hosting site web pages each with a respective appearance.

It is another object of the invention to reuse web-based content without requiring a content provider web site to modify content or install special purpose software.

It is another object of this invention to enable a publisher of an electronic document to control the reformatting of the document by a hosting site.

These objects and others are accomplished by managing copyrighted content on the Internet and World Wide Web by means of a filtering and formatting service located on a hosting server. The invention provides an automated system



6,128,655

3

for replicating published web content and associated advertisements in the context of a hosting web site. At the hosting web site, the invention includes the process of brokering a client browser's request for a web page, analyzing the returned content and splitting it into component elements, extracting the desired component elements, recasting the desired elements in the look and feel of the hosting site and sending the recast content to the requesting client as a web page. Once the reformatted file is received at the client, the client browser interprets the HTML in the web page, presenting the content in the context of the hosting web site. On the content provider's web site, the details of the transaction in the web server logs are preserved, proxying a direct page view and ad impression.

The foregoing has outlined some of the more pertinent objects and features of the present invention. These objects should be construed to be merely illustrative of some of the more prominent features and applications of the invention. Many other beneficial results can be attained by applying the disclosed invention in a different manner or modifying the invention as will be described. Accordingly, other objects and a fuller understanding of the invention may be had by referring to the following.

#### BRIEF DESCRIPTION OF THE DRAWINGS

For a more complete understanding of the present invention and the advantages thereof, reference should be made to the following Detailed Description of the Preferred Embodiment taken in connection with the accompanying drawings in which:

FIG. 1 is a representative system in which the present invention is implemented.

FIG. 2 is a simplified block diagram of a requesting client, hosting server and plurality of content provider servers which illustrates an overview of the process of the present invention.

FIG. 3 is an illustrative example of an unchanged source web page as it would normally be presented by a client browser as retrieved from the content provider web server.

FIG. 4 is an illustrative example of the reformatted web page as presented at the client browser after having undergone the processing of the present invention.

FIGS. 5A and 5B are more detailed flowcharts of a preferred method of the processes which occur at the hosting server.

FIG. 6 is pictorial representation of a hosting filter definition interface.

FIG. 7 is a block diagram of the major components of the data processing system unit on which the invention may be practiced.

#### DETAILED DESCRIPTION OF THE DRAWINGS

A representative system in which the present invention is implemented is illustrated in FIG. 1. A plurality of Internet client machines 10 are connectable to a computer network Internet Service Provider (ISP) 12 via a network such as a dialup telephone network 14. As is well known, the dialup telephone network usually has a given, limited number of connections 16a-16n. ISP 12 interfaces the client machines 10 to the remainder of the network 18, which includes the hosting server 19 and a plurality of web content provider server machines 20. A client machine typically includes a suite of known Internet tools, including a Web browser 13, to access the servers of the network and thus obtain certain services. These services include one-to-one messaging

4

(e-mail), one-to-many messaging (bulletin board), on-line chat, file transfer and browsing. Various known Internet protocols are used for these services. Thus, for example, browsing is effected using the Hypertext Transfer Protocol (HTTP), which provides users access to multimedia files using Hypertext Markup Language (HTML). The collection of servers that use HTTP comprise the World Wide Web, which is the Internet's multimedia information retrieval system.

As shown in FIG. 2, the invention is a method and system for extracting Web-based content, especially, but not limited to, Web-based news articles, from content provider or source Web sites for use by the hosting or "pass-through" Web site. These articles typically are revenue-generating content for the publisher by carrying advertising banners above and/or below the article text. Therefore, the publishers must benefit from the arrangement provided by the hosting site to be interested in licensing their content for a low or no fee. As explained below, the web content provider maintains his ad revenue as the number of "hits" on the advertisements are maintained in a transparent manner. As the articles are also posted at the hosting site, ad revenues can actually increase since the ad impressions are being solicited from two sites rather than one.

During configuration, the pass through publisher 101 at the hosting site 103 is provided with the URLs 105 for the desired content provider web servers 107 and a set of filters 109 for the content publisher's document templates 111. For ease in illustration, a single client 113 and a single web content server 107 are depicted. However, the reader should understand that a plurality of clients and web content servers are typically interconnected through the agency of the hosting site. Upon a request 115 from a client 113 for a given web page, typically made through an HTTP request from the resident browser, the process for providing a page using the pass through mechanism begins. Next, after having established that the requested page originates at the web content server 107, the hosting site makes a request 117 for the page. Presuming that this is a first request for the web page or that a more up to date version of the page is available at the web content provider than is cached locally, the page is returned 119. In today's web technology, the web page is typically an HTML file with references to the component .wav, .mov, gif and JPEG files which together make up the web page as perceived by the user. Secondary page components such as cascading style sheets and Java applets can also be accommodated by the invention. The list above is merely exemplary; any component on a web page can be extracted and recast into the hosting site context by the present invention.

Next, the pass through publisher 101 retrieves the filter definitions and policies from the filter database 109 for this particular content provider web site. Using the filters and the retrieved HTML page, the pass through publisher 101 parses the HTML source for desired components of the page. Typically, this is the title of the article, the ad banner or banners and the article text itself, although other items on the page are potentially desirable. These pieces of content are then recast into a new web page by means of an HTML template 121 that matches the look and feel of the hosting Web site. The new page includes the graphics of the hosting provider as well as the navigational features of the hosting site. This page is then sent 123 to the client 113 for presentation by the browser. In a typical web interaction between browser and server, once the browser receives the HTML page, it issues additional requests for the component files such as .gifs, e.g., ad banners. For the ad banners themselves, the new page preserves the call 125 back to the

6,128,655

5

content provider so that the correct advertising content is presented. It is common that each request of a web page from a server can be refreshed with a different advertisement.

In this way, the end user receives a page with graphic and navigation features from the hosting Web site that has an embedded article from the publisher and an advertisement served from the publisher's site. The final result is content viewed by the end user in host site's native Web context, with an ad banner served from the original publisher, thereby preserving their revenue stream.

It should also be noted that the article text is preferably cached in a local cache **131**, on the hosting Web server **103**, for faster access and guaranteed access in the event that the publisher's Web site becomes inaccessible. The invention encompasses several variations in the types of information parsed from the page and cached locally. Some of this information may be incorporated in the recast HTML page and some may be used for version checking. For example, information in the HTML header such as "last modified", "content length" and "content type" could be kept with the article text so that the copy in the cache can be compared to the version available at the content provider site. However, in the preferred embodiment, the applicants have found it to be more efficient to simply compare the "last modified" data in the HTML header with the "last modified" data in the hosting system's cache file. Remember that the hosting site **103** makes the request **117** for the client to preserve the accounting data for the content provider web site **107**. Since the header data is among the first to be transmitted **119** in response, after a simple compare establishes that the cached version and the version currently available at the content provider web site are the same, the transmission **119** from the content provider can be ended. The hosting system **103** then uses the cached copy of the article. In the event of no response from the content provider web site, a cached copy of the article is used. When there is no cached copy of an article, or the compare establishes that a more recent version of the article is available, the entire transmission **119** from the content provider is received for processing. Alternatively, rather than waiting for a client request, the 'freshness' of the cached content can be ascertained by automatically generating HTTP requests from the cached URLs and monitoring data in the HTTP headers when the page is hit in the background, updating the cache any time the web content provider changes their data.

The aim of caching pass-through web content is to maximize efficiency by minimizing network bandwidth requirements while preserving the transparency of the transaction. By caching copies of the parsed content on the hosting server, serving the content to the end user directly and simulating their 'hit' on the publisher's site in the background, the end user gets content directly from hosting site without having to wait for data to travel from the content web provider's site to the hosting site. However, this method only assures a correct count for the web content provider whose advertising systems use a secondary HTTP request for the image retrieval to generate the ad impression. For systems that rely on dynamic HTML generation to log ad impressions, the ad content must be retrieved for each user and not cached on the host site. The static portion of the page, i.e. the article, however, can be cached, since it remains the same for each visit at least for a relatively long period of time. Serving the recast page to the end user will be delayed by the network for retrieving the ad content, but if the publisher's site becomes unavailable, the end user will not be affected.

6

An alternative embodiment to the invention is to provide a client based Java applet that retrieves dynamic content from the web content provider's server directly from the end user's browser. This allows the recast page to be loaded from the hosting site's cache to the client browser and invoking the Java applet for the retrieval of marked dynamic content. This reduces the network bottleneck at the hosting site for dynamic HTML ad generation.

Before describing the hosting process in greater detail, the reader's attention is directed to FIGS. **3** and **4** which respectively show the appearance of a content provider web page as originally sent and the recast web page as sent from the hosting site. It should be understood that the page in FIG. **3** is never actually displayed by the client browser, however, showing the page as it would have been presented if the client had made the request directly to the content provider web site is useful to understand the principles of the invention.

As shown in both figures, the browser window **201** bounds each web page and contains standard graphical user interface elements such as title bars, menu items and scroll bars. The browser shown is Netscape Communicator, showing that a standard client browser can be used unmodified to practice the invention. In the client area **203** showing the unmodified page, the logo banner **205**, title area **207** and article text **209** are shown. Under the logo banner **205**, a set of links **211** will retrieve other pages from the content provider server. Finally, at the bottom of the page, an ad banner **213** is presented.

In FIG. **4**, the recast page is shown in client area **303**. In this example, the logo banner **305** is preserved, but moved to a new location (centered). The title area **307** and article text **309** have changed location, font and font size and line length. Other format changes are possible. Some, but not all of the links **311** to other content provider web pages have been preserved according to the policy for the web content provider. Since these links may be important to the web content provider to generate additional hits for other advertising revenue, the provider may wish to institute a policy that at least some of these links will be preserved in the recast page. The ad banner **313** appears at the bottom of the page. Note also that navigational features **315** and **317** native to the hosting server have been added to the page. A background border **319** giving the hosting web site a distinctive look and feel has also been added. Of course, those skilled in the art will recognize that the examples of "desired content" are merely exemplary. The example of the top ad, article and bottom ad is common to many web news articles. The invention allows the hosting site to extract and recast any number or type of desired content elements from the web content provider page.

Depending upon the policy for the web content provider, variations in which elements are preserved in the recast page are possible. For example, the logo **305** is an optional feature. It may be removed or reduced in size or replaced by a different logo stored in the filter definition. The links **311** are optional; they could be removed, reformatted or relocated. As a technical matter, the ad banner **313** is optional, however, from a practical standpoint to obtain content at a low licensing fee, they are probably mandatory. Other items such as copyright notices are not shown in the figure, but could be preserved.

The process by which a new page is registered into the hosting system is depicted in FIG. **5A**. It begins in step **401**, when a new page or some other registration action is detected. Step **403** determines whether the page is from an

6,128,655

7

existing account, i.e. an existing web content provider web site. If not, a new account is started step 405. The account or folder is a convenient place to store filter definitions, policies and any transaction information which pertains to a particular content provider.

The test in step 407 determines whether it is a new page, either because of a new URL or new version, which has started the registration process. If it is not a new page, step 409, determines whether it is a request to create or change a filter definition which has started the registration process. For the purposes of this diagram, the policy for a content provider is considered part of the filter definitions although the information can certainly be kept in a separate file. The process will exit in step 411 if there is no filter definition to change.

In step 413, it is determined whether there is a suitable filter definition in the account folder for the content provider for the new page. As most pages in a web site share a common format and style, it is envisioned that a relatively small set of filter definitions can be used for all of the pages from a particular site. If there is no existing filter definition suitable, in step 415, a new filter definition is created for the page. There is more discussion on the creation of filter definitions and policies below in connection with FIG. 6.

In step 417, the page, i.e. URL is associated with the appropriate filter definition and in step 419 the appropriate changes to the account, URL and filter definition files are made. Optionally, the new page can be processed and cached as part of registration. Thus, in step 421, the filter definition is used by the pass through publisher to extract the desired portions of the page. In step 423, these portions of the page are cached for retrieval in the event of a client request. The process ends, step 425.

In FIG. 5B, the process for parsing and reusing web content by the pass through publisher is shown. When a client requests a new document from the pass through publisher at the hosting web site, the requesting web client information is recorded, and a request is made by the hosting web site to the content provider's web server on behalf of the requesting web client. The HTTP request to the web content provider server is similar to that which the requesting client could make to the content provider site directly, except with the hosting site as the originator. This assures that the web content server's log files record a visit by the requesting client which is essential for preserving the content provider's revenue stream.

As mentioned above, the hosting site preferably caches content likely to be requested by a client to improve the speed and reliability of the hosting web site pages. In this way, if the document has not changed since the pass through publisher last polled the site, it is retrieved from the local cache after registering the "hit" on the remote server. This reduces Internet bandwidth requirements and improves performance on both the hosting web server and the web content provider server.

However, for the process depicted in FIG. 5B, new content has been retrieved from the web content provider web server, step 451. Once the document content has been retrieved from the host provider, the filter database is searched for the appropriate filter definition, step 453, the filter definition kept for the web content provider. The information in the filter definition will help the pass through publisher parse the document structure of the web page, extracting the desired information. In step 457, a test is performed to determine whether the parsing was a success.

If a filter definition for the page or web content provider is not found, or the first attempt using the associated filter

8

definition was not a success, the pass through publisher can fall back to a series of default filters which will assist in parsing the data, step 459. The hosting site will still be able to present the reformatted content, however, the process will not be as efficient as through an existing filter definition. This "best guess" approach utilizes several methods, including looking for common references to advertising engines, etc. As discussed below, the publisher can also look for a set of embedded tags indicating the desired content. Any document that a filter can not be found for can be logged, allowing staff to later create appropriate filter definitions. In practice, however, hosting sites employing the pass through technique will be able to define templates appropriate to all "rehosted" content. Most content provider sites employ a standard look and feel in their documents, allowing for filters that are appropriate for large numbers of documents found on a particular web site, if not every document on the entire provider web site.

These excerpted components are then run through the pass-through publisher's "post-processing" system to assure that they do not contain "dangerous" formatting code fragments that could adversely effect the hosting web site, step 461. For example, when articles are extracted from within a TABLE structure, HTML TABLE fragments could be left in the filtered HTML that could destroy formatting on the hosting web site. As another example, interactive or browser dependent scripting code could be found in the filtered HTML that may not make sense in the document's new context. The post filtering tasks should also include fixing any relative URLs embedded in the original web page to preserve their original function. Optionally, this can be accomplished by pointing the URLs to the hosting site for handling. For example, many documents are split into several pages by the web publisher. The link to the next part of the article can be translated to a hosting site link so that the next part is automatically served in the hosting site's context. The relative link could also be translated to an absolute link so that it will still lead to the content provider server even when selected in the recast page. As would be readily understood by those skilled in the art, these post filtering tasks could easily be performed by one of the filters, however, the applicants have found it to be convenient to separate the tasks thus simplifying the construction of the filter definitions.

The component HTML file, once extracted, separated, and post filtered is then reformatted into a new document in the style and context of the hosting web site, step 463. This is done by another component of the pass through publisher, a web publishing application that creates a "dynamic publishing template". The web publisher injects the excerpted content, titles, copyright statements and logos as received from the post filtering process. In step 465, the desired components are cached, which may include components useful in determining the version of a web page, but are not used in the recast page. In step 467, the recast page is sent to the requesting client. The process ends, step 469. Once presented by the requesting browser, the content of the hosting web site appears seamless to the user, although it may originate at a plurality of web content provider sites as well as the hosting site itself.

Since the code from the original content has been abstracted and separated from its style and formatting, it is now possible to format before sending it to the user in any of a variety of styles. This can prove useful in a variety of situations. It is common for the web sites of several smaller organizations to be "hosted" by an organization with the technical expertise and capital equipment allowing the



6,128,655

9

smaller organizations to concentrate on creating the content for the web sites rather than the details of maintenance of the server machines. A single pass through publisher could provide a different look and feel for each of the different organizations hosted on its web servers. Alternatively, a single hosting web site could provide several different alternative formats. The choice of which format to present to a particular user could be based on the organization or location associated with the user. Alternatively, the web site could allow the user to choose from among the different formats based on a registration of his preferences in a user profile. Thus, the look and feel of a web site can change dependent upon the requesting audience.

The invention provides a mechanism which allows a hosting web site to provide a wide variety and great amount of third party Web content without incurring high licensing costs. Another benefit of the pass through system is in cost savings. Unlike a traditional system of licensing and republishing content, the hosting system does not require a large production staff since the republishing and re-styling of the content is automatic. A hosting system can provide a much faster production cycle and assure that the content does not quickly go "out of date".

A discussion of filter definition creation follows. The collection of document filters help the pass through engine understand the structure of a wide variety of web documents. The document filters can be created through several methods, including the analysis of the HTML source code, imbedded comments or delimiters and through comparisons with similar documents. Once the style of the web site is understood, a filter can be developed to look for the portion of the original document in which the hosting site is interested in reformatting. Inconsistencies in document style or structure can be neutralized by the use of custom code imbedded in the web page and detailed in the filter definition.

A CGI or other program can be used to create filter definition files. FIG. 6 shows a user interface in which tags or text can be entered manually so that the pass through publisher can more easily parse a web content provider's web pages. In the browser window **501**, client area **503** contains a plurality of controls for a set of desired components. Entry fields **505**, **507**, **509**, **511**, **513**, **515**, **517**, **519** and **521** are respectively used to enter the filter name, the logo name, a copyright string, a beginning of the top banner ad, the ending of top banner ad, the beginning of the article text, the ending of the article text, the beginning of the bottom ad and the ending of the bottom ad. Note that certain items such as logo name and copyright string could be replacements for those which occur in the web page, rather than indicators of the desired content.

A set of check boxes **523** allows the filter designer to indicate which of these items he wishes to keep on the recast page. The table stripping check boxes **525** indicate whether table formatting should be stripped from certain areas of the content provider's page. Custom filter code can be entered in field **527**. Field **529** allows the entry of custom code for filtering code behaviors outside the predefined filters. Special cases can be accommodated by adding a function in Perl, Java, JavaScript or a specialized filter scripting language. Push button **531** allows the user to change to a different filter definition.

Each filter definition is stored in a filter definition database accessible by the pass through publisher. The publisher uses the filter definition to break the content into component parts: The title area, primary and secondary advertisements,

10

and the content itself. The title area includes the title of the web page and is typically marked by HTML tags. The primary and secondary advertisements usually occur at the top and bottom of the web page, but may be located at different locations. They are typically marked in the HTML by tags or comments indicating an advertisement. Depending on various factors, such as the desired look and feel for the hosting web site, the cross-publishing agreement with the content provider, i.e. allowing for republishing certain types of web content but not others and the filter, the content may be very plain. A "bare bones" filter may strip out any extraneous links or "side bars" of information. Alternatively, the content may be a verbatim copy of a selected portion of the original web page.

In addition to providing the system with information on separating the components of the document, filter definitions also include publisher specific information such as the logo or copyright statements and policies that should be used by the pass through publisher when formatting the new version of the document.

Alternatively, the logo and copyright statements could be excerpted components like the title, ads and content.

The filter definitions can also include the "policy" for a particular web content provider. Any number of policies can be established based on publisher, article, article section or any other distinguishing criteria that can be identified. Policies might govern whether content is licensed for use on an intranet, but not on the Internet, or vice versa, or both; how many times a document may be served off a host site; whether the publisher's ads should be passed through or not; what kind of caching strategy should be applied; what cost each view of the article carries for the host site; and so on. The specific types of policies available will depend on the context in which pass-through is being used, whether as a commercial product, integrated into custom solutions, or bundled with other products.

The client machine may be a personal computer such as a desktop or notebook computer, e.g., an IBM or IBM-compatible machine running under the OS/2® operating system, an IBM ThinkPad® machine, or some other Intel x86 or Pentium®-based computer running Windows '95 (or the like) operating system. Of course, the invention may be run on a variety of computers or collection of computers under a number of different operating systems. The computers on which the client software and the hosting and content provider web site reside could be, for example, a personal computer, a mini computer, mainframe computer or a hand held computer. Although the specific choice of computer is limited only by processor speed and disk storage requirements, it is typical that the client computer will be somewhat "lighter weight" than the web server computers. For example, computers in the IBM PC series of computers could be used as clients in the present invention. One operating system which an IBM personal computer may run is IBM's OS/2 Warp 4.0. For the web servers, the computer system might be in the IBM RISC System/6000 (TM) line of computers which run on the AIX (TM) operating system.

In FIG. 7, a computer **710**, comprising a system unit **711**, a keyboard **712**, a mouse **713** and a display **714** are depicted in block diagram form. The system unit **711** includes a system bus or plurality of system buses **721** to which various components are coupled and by which communication between the various components is accomplished. The microprocessor **722** is connected to the system bus **721** and is supported by read only memory (ROM) **723** and random access memory (RAM) **724** also connected to system bus

6,128,655

11

721. A microprocessor in the IBM PC series of computers is one of the Intel family of microprocessors including the 386, 486 or Pentium microprocessors. However, other microprocessors including, but not limited to, Motorola's family of microprocessors such as the 68000, 68020 or the 68030 microprocessors and various Reduced Instruction Set Computer (RISC) microprocessors such as the PowerPC chip manufactured by IBM might be used by the present invention. Other RISC chips made by Hewlett Packard, Sun, Motorola and others may be used in the specific computer.

The ROM 723 contains among other code the Basic Input-Output system (BIOS) which controls basic hardware operations such as the interaction of the processor with the disk drives and the keyboard. The RAM 724 is the main memory into which the operating system and application programs are loaded. The memory management chip 725 is connected to the system bus 721 and controls direct memory access operations including, passing data between the RAM 724 and hard disk drive 726 and floppy disk drive 727. The CD ROM drive 732 also coupled to the system bus 721 is used to store a large program or amount of data, e.g., a multimedia program or presentation.

Also connected to this system bus 721 are various I/O controllers: The keyboard controller 728, the mouse controller 729, the video controller 730, and the audio controller 731. As might be expected, the keyboard controller 728 provides the hardware interface for the keyboard 712, the mouse controller 729 provides the hardware interface for mouse 713, the video controller 730 is the hardware interface for the display 714, and the audio controller 731 is the hardware interface for the speakers 715. An I/O controller 740 such as a Token Ring Adapter enables communication over a network 746 to other similarly configured data processing systems.

One of the preferred implementations of the invention is as sets of instructions 748-752 resident in the random access memory 724 of one or more computer systems configured generally as described above. Until required by the computer system, the set of instructions may be stored in another computer readable memory, for example, in the hard disk drive 726, or in a removable memory such as an optical disk for eventual use in the CD-ROM 732 or in a floppy disk for eventual use in the floppy disk drive 727. Further, the set of instructions can be stored in the memory of another computer and transmitted in a transmission means such as a local area network or a wide area network such as the Internet when desired by the user. One skilled in the art knows that storage or transmission of the computer program product changes the medium electrically, magnetically, or chemically so that the medium carries computer readable information.

Further, the invention is often described in terms that could be associated with a human operator. While the operations performed may be in response to user input, no action by a human operator is desirable in any of the operations described herein which form part of the present invention; the operations are machine operations processing electrical signals to generate other electrical signals.

As used herein, "Web client" should be broadly construed to mean any computer or component thereof directly or indirectly connected or connectable in any known or later-developed manner to a computer network, such as the Internet. The term "Web server" should also be broadly construed to mean a computer, computer platform, an adjunct to a computer or platform, or any component thereof. Of course, a "client" should be broadly construed to

12

mean one who requests or gets the file, and "server" is the entity which downloads the file. Moreover, although the present invention is described in the context of the Hypertext Markup Language (HTML), those of ordinary skill in the art will appreciate that the invention is applicable to alternative markup languages including, without limitation, SGML (Standard Generalized Markup Language), dynamic HTML and XML (Extended Markup Language).

Moreover, while the preferred embodiment is illustrated in the context of a dialup network and the Internet, this is not a limitation of the present invention. The invention can also be implemented in an intranet environment where a large organization may have several content provider units which provide content for content using units which target different customer segments and have different trade identities. Thus, while the content using units may utilize much of the same information, each will want to recast the information in a different look and feel to project their own trade dress.

There are many possible approaches to creating parsing filters for the invention. For predictable sets of documents the approaches are straight forward. Where the documents vary a great deal some intelligence in either the pass-through mechanism or the user who is configuring the filter is required. In other words, either a user needs to customize the filtering at a level nearing programming or scripting skills to account for all the possible variations after study of a sample set of documents, or the pass-through mechanism needs to be imbued with some level of fuzzy logic or artificial intelligence.

If an agreed on set of tags used by the web content provider and hosting sites, 100% of Web documents are parseable. Thus, no intelligence is required from the pass through mechanism and no programming or scripting is required of the user. Special tags are embedded in the source of the targeted document(s) which identify the content areas. This allows a 'default' filter to be used that requires no customization beyond supplying it with the target URL. These special tags could take the form of HTML comments. In the future, the tags can be formalized as an XML Document Type Definition. It is envisioned that HTML editing programs used by the content provider can add the tags as the web content is created automatically.

The speed of document retrieval is an issue with the invention, since in essence a single user's request for a document is transformed into two separate requests, with all the potential for bottlenecks that any Web transaction has. Caching can provide a partial solution, the title area, article body and other desired content can be cached locally on the hosting site, so that it can be delivered to the user more quickly. Ad source needs to be retrieved from the source site on a per-user basis to preserve the ad accounting process of many web sites. In addition, many ad systems serve ads based on the visitor's browser or other information.

The invention can be configured a stand alone server software product. This would resemble a proxy server and would serve two purposes: it would help the speed issue by devoting more resources to the hosting activity, and it would allow the servicing of several hosting web sites from a single server.

The invention solves several business and technical problems. It provides an attractive mechanism to obtain permission to reprint Web-based content with little or no licensing fees. Since the original publisher's transaction records are preserved, their existing revenue base is maintained through the number of ad impressions counted. Since the ad impressions are now also occurring on the hosting web site with

6,128,655

13

very little work on the part of the original publisher, the revenue is very likely to be increased. Thus, increased traffic is generated for both the hosting web site as well as the content provider's site with very little manual intervention after configuration.

The invention is very flexible and is easily configured to accommodate a wide variety of web content. Through the use of document templates and standard filters, the invention allows simple modification of these elements to tailor them to any number of different content providers' formats and document templates. Once the hosting web server has been configured for a set of content providers, the production staff necessary to republish articles is minimal. Content can be extracted without the content provider web site modifying content to a special format or installing special purpose software. Articles in the hosting web site are automatically synchronized with those in the content provider as changes are made at the content provider web site (so long as noncached material is used). By abstracting the content from any particular content provider site and reformatting the content to the hosting site's format a consistent look and feel is maintained.

In one preferred embodiment of the invention, the hosting web server caches content locally to speed delivery to the requesting client and minimize dependency on the content

14

provider web site. In other embodiments of the invention, unauthorized requests are blocked, eliminating a potential avenue for abuse of the system and copyright violation.

5 In the attached appendix, examples are given of a content provider's original web page, the template in which in hosting site inserts the excerpted desired content and the resulting recast page with comments. These examples will help the reader more fully understand the principles of the present invention.

10 While the invention has been shown and described with reference to particular embodiments thereof, it will be understood by those skilled in the art that the invention can be practiced, with modification, in other environments. For example, although the invention described above can be conveniently implemented in a general purpose computer selectively reconfigured or activated by software, those skilled in the art would recognize that the invention could be carried out in hardware, in firmware or in any combination of software, firmware or hardware including a special purpose apparatus specifically designed to perform the described invention. Therefore, changes in form and detail may be made therein without departing from the spirit and scope of the invention as set forth in the accompanying claims.

---

#### APPENDIX

---

Original Content Provider HTML:

IBM Global Services

<http://www.ibm.com/services/articles/whatwedo.html>What we can do for you<http://www.ibm.com/services/business/>Viewpoints<http://www.ibm.com/services/career/>Careers<http://www.ibm.com/services/business/feature.html>Case Studies<http://www.ibm.com/services/pressrel/>News<http://www.ibm.com/services/navtools/otherservices.html><http://www.ibm.com/Search>Search<http://www.ibm.com/services/profservices/index.html>Professional Services<http://www.as.ibm.com/>Product Support Services<http://www.ibm.com/globalnetwork/>Network Services<http://www.ibm.com/services/ourportfolio.html>Our Portfolio IBM Announces New e-business Services for Security Builds on popular packaged e-business services offerings March 24, 1998

BOSTON, Massachusetts, March 24, 1998 ... IBM today announced new global security services that build on the company's portfolio of e-business services introduced last October. IBM's e-business offerings help business use networks and Internet technologies to more securely buy and sell on the Web and improve internal and external communication. IBM made these announcements at Internet Commerce Expo.

<./ebus/security.html>IBM Security Services help customers of all sizes assess

and improve security in their computing environments. They address exposures across operations, including policy and management systems, applications, networks, systems and physical site security. IBM has the unique capability as a security services provider to give customers a choice of individual offerings or a comprehensive, end-to-end security solution.

.....

\*IBM is a registered trademark of International Business Machines Corporation

<http://www.ibm.com/> IBM Homepage <http://www.ibm.com/Orders/> Order

<http://www.ibm.com/Assist/> Contact IBM

<http://www.ibm.com/IBM/Employment>

Employment <http://www.ibm.com/Privacy/>Privacy <http://www.ibm.com/Legal/> Legal

The Hosting Site Web Page Template

Home

<http://dev2.cross-site.com/apps/top.map> Need Help? Click on the '?'

<http://dev2.cross-site.com/apps/side.map> Need Help? Click on the '?'

<http://dev2.cross-site.com/cs/?section=News&text=news/news.html>News |

<http://f2.dejanews.com/crosssite/>Forums |

<http://dev2.cross-site.com/cs/?section=Columns&text=columns/columns.html>C  
olumns |

<http://dev2.cross-site.com/cs/?section=Resources&text=resources/resources.  
html>Resources |

6,128,655

15

16

-continued

## APPENDIX

```

<http://dev2.cross-site.com/cs/?section=Downloads&text=downloads/downloads.
html>Downloads |
<http://dev2.cross-site.com/cs/?section=Cross-Site&text=about/about.html>Abo
ut |
<http://dev2.cross-site.com/cs/?section=Products&text=products/products.htm
l&sidebar=products/sidebar.html>Products |
<http://dev2.cross-site.com/cs/?section=Employment&text=employment/employem
t.html>Employment
<http://dev2.cross-site.com/cs/?sidebar=home/sidebar.html>Home |
<http://dev2.cross-site.com/cs/?section=Search&text=sitesearch/search.html&t
itle=Search&logo=logo.crosssite>Search |
<http://dev2.cross-site.com/cs/?section=Mail&text=mail/mail.html>Email |
<http://dev2.cross-site.com/cs/?section=Contact&text=about/contact.html>Cont
act
| <http://dev2.cross-site.com/cs/?section=Help&text=support/help.html>Help
(C)1998 Tivoli Systems
The Recast Web Page (including comments):
(The parsing engine extracted this code from the URL):
<IMG SRC="http://www.ibm.com/services/images/animh.gif" alt="IBM Global
Services" WIDTH=584 HEIGHT=54 BORDER=0><br>
<TABLE WIDTH=584 CELSPACING=0 CELLPADDING=0 BORDER=0>
<TR><TD><NOBR><A
HREF="http://www.ibm.com/services/articles/whatwedo.html"
TARGET=_top><IMG SRC="http://www.ibm.com/services/images/foryou3.gif"
ALT="What we can do for you" WIDTH=145 HEIGHT=18 BORDER=0></A><A
HREF="http://www.ibm.com/services/business/" TARGET=_top><IMG
SRC="http://www.ibm.com/services/images/viewpt3.gif" ALT="Viewpoints"
WIDTH=81 HEIGHT=18 BORDER=0></A><A
HREF="http://www.ibm.com/services/career/" TARGET=_top><IMG
SRC="http://www.ibm.com/services/images/careers3.gif" ALT="Careers"
WIDTH=67 HEIGHT=18 BORDER=0></A><A
HREF="http://www.ibm.com/services/business/feature.html" TARGET=_top><IMG
SRC="http://www.ibm.com/services/images/casestdy3.gif" ALT="Case Studies"
WIDTH=90 HEIGHT=18 BORDER=0></A><A
HREF="http://www.ibm.com/services/pressrel/" TARGET=_top><IMG
SRC="http://www.ibm.com/services/images/news3.gif" ALT="News" WIDTH=52
HEIGHT=18 BORDER=0></A><A
HREF="http://www.ibm.com/services/navtools/otherservices.html"><IMG
SRC="http://www.ibm.com/services/images/countrysites.gif" WIDTH=87
HEIGHT=18 BORDER=0></A><A HREF="http://www.ibm.com/Search"
TARGET=_top><IMG
SRC="http://www.ibm.com/services/images/search3.gif" ALT="Search"
BORDER=0></A></NOBR></TD></TR>
</TABLE>
(It then inserted the code into the hosting site's template, thusly:)
<CENTER>
<TABLE BORDER=0>
<TR>
<TD>
<IMG SRC="http://www.ibm.com/services/images/animh.gif" alt="IBM Global
Services" WIDTH=584 HEIGHT=54 BORDER=0><br>
<TABLE WIDTH=584 CELSPACING=0 CELLPADDING=0 BORDER=0>
<TR><TD><NOBR><A
HREF="http://www.ibm.com/services/articles/whatwedo.html"
TARGET=_top><IMG SRC="http://www.ibm.com/services/images/foryou3.gif"
ALT="What we can do for you" WIDTH=145 HEIGHT=18 BORDER=0></A><A
HREF="http://www.ibm.com/services/business/" TARGET=_top><IMG
SRC="http://www.ibm.com/services/images/viewpt3.gif" ALT="Viewpoints"
WIDTH=81 HEIGHT=18 BORDER=0></A><A
HREF="http://www.ibm.com/services/career/" TARGET=_top><IMG
SRC="http://www.ibm.com/services/images/careers3.gif" ALT="Careers"
WIDTH=67 HEIGHT=18 BORDER=0></A><A
HREF="http://www.ibm.com/services/business/feature.html" TARGET=_top><IMG
SRC="http://www.ibm.com/services/images/casestdy3.gif" ALT="Case Studies"
WIDTH=90 HEIGHT=18 BORDER=0></A><A
HREF="http://www.ibm.com/services/pressrel/" TARGET=_top><IMG
SRC="http://www.ibm.com/services/images/news3.gif" ALT="News" WIDTH=52
HEIGHT=18 BORDER=0></A><A
HREF="http://www.ibm.com/services/navtools/otherservices.html"><IMG
SRC="http://www.ibm.com/services/images/countrysites.gif" WIDTH=87
HEIGHT=18 BORDER=0></A><A HREF="http://www.ibm.com/Search"
TARGET=_top><IMG
SRC="http://www.ibm.com/services/images/search3.gif" ALT="Search"
BORDER=0></A></NOBR></TD></TR>
</TABLE>
</TD>
</TR>

```



-continued

APPENDIX

```
</TABLE>
</CENTER>
<A NAME="#TOP"></A>
<FONT SIZE="+1" COLOR="#000099" FACE="Arial, Helvetica">
<B>News<B>
</FONT>
<!-- START TOP NAV BUTTONS -->
<TABLE CELLPADDING=0 CELLSPACING=0 BORDER=0 WIDTH=100%>
<TR ALIGN=RIGHT VALIGN=TOP>
<TD BGCOLOR=FFCC33 ALIGN=RIGHT VALIGN=CENTER BORDER=0
WIDTH=100% COLSPAN=2>
<A HREF="http://dev2.cross-site.com/apps/top.map">
<IMG NAME="topbuttons" HEIGHT=35 WIDTH=175
SRC="http://dev2.cross-site.com/images/topbuttons.gif"
BORDER=0 ALT="Need Help? Click on the "?" ISMAP ></A>
</TD>
</TR>
<!-- END TOP NAV BUTTONS -->
(Similarly, the template has this insertion spot for the article from the
content provider's document:)
<TABLE BORDER=0>
<TR>
<TD>
</TD>
</TR>
</TABLE>
(Into which the extracted article is inserted:)
<H3>
IBM Announces New e-business Services for Security
<BR><SMALL>Builds on popular packaged e-business services
offerings</SMALL>
</H3>
<P><B>March 24, 1998</B></P>
<P>BOSTON, Massachusetts, March 24, 1998 ... IBM today announced new global
security services that build on the company's portfolio of e-business
services introduced last October. IBM's e-business offerings help business
use networks and Internet technologies to more securely buy and sell on the
Web and improve internal and external communication. IBM made these
announcements at Internet Commerce Expo.
<p> <a href=../ebus/security.html>IBM Security Services</a> help
customers of all sizes assess and improve security in their computing
environments. They address exposures across operations, including policy
and management systems, applications, networks, systems and physical site
security. IBM has the unique capability as a security services provider to
give customers a choice of individual offerings or a comprehensive,
end-to-end security solution.
...
<BR>
</FONT>
</TD>
</TR>
</TABLE>
(The end result is a unified HTML document with elements from the
publisher's page inserted into the host site's template to create a
seamless whole.)
```

- We claim:

1. A method for recasting web content on a hosting site, comprising the steps of:

  - responsive to a request from a client browser for a recast web page from a hosting web server, generating a request by the hosting web server for an original web page from a content provider web server;
  - parsing the original web page for a first set of desired content elements;
  - inserting the first set of desired content elements into a web page template containing a hosting web server format, thus creating the recast web page; and
  - serving the recast web page to the client browser;

wherein the appearance of the recast page when presented by the client browser is as though all elements originated at the hosting web server.
2. The method as recited in claim 1, wherein one of the desired content elements is an advertisement element from the content provider web server, and the method further comprises the step of inserting a call back to the content provider web server for the advertising element.

3. The method as recited in claim 1 further comprising the steps of:

  - caching the desired content from the original page at the hosting web server;
  - responsive to a second request for the recast page from a client browser, determining whether there is a more recent version of the original page at the content provider server; and
  - using the cached desired content if there is no more recent version of the original page to respond to the second request.

6,128,655

19

4. The method as recited in claim 3 further comprising the steps of:

    parsing the most recent version of the original web page for the first set of desired content elements;

    inserting the first set of desired content elements into a web page template containing a hosting web server format, thus creating a new version of the recast web page;

    serving the new version of the recast web page to the client browser responsive to the second request; and  
    caching the desired content elements from the more recent version of the original page at the hosting server.

5. The method as recited in claim 3 wherein the cached desired content elements are cached in the form of the recast page.

6. The method as recited in claim 1 further comprising the steps of:

    searching for a filter definition for web pages from the content provider server in a store of filter definitions; and

    responsive to identifying a filter definition for the content provider, using the identified filter definition to parse the original page.

7. The method as recited in claim 1, further comprising the steps of:

    searching for a filter definition for web pages from the content provider server in a store of filter definitions; and

    responsive to a failure to find a filter definition for the content provider, using a default filter to parse the original page.

8. The method as recited in claim 1, wherein at least some of the content in the recast page originates at a second content provider server and the method further comprises the steps of:

    responsive to a request from a client browser for a recast web page from a hosting web server, generating a request by the hosting web server for a second original web page from the second content provider web server;  
    parsing the second original web page for a second set of desired content elements; and  
    inserting the first and second set of desired content elements into a web page template containing a hosting web server format, thus creating the recast web page; wherein when presented by a client browser, both the first and second set of desired content elements look native to the hosting server.

9. The method as recited in claim 1, wherein the web page template contains navigational features for the hosting server.

10. The method as recited in claim 1, further comprising the step of processing the desired content elements to eliminate harmful code, prior to insertion in the web page template.

11. The method as recited in claim 1, further comprising the step of revising relative links in the original page to links appropriate to the recast web page.

12. The method as recited in claim 1, further comprising the steps of:

    responsive to a second request from a client browser for a second recast web page from a hosting web server, generating a request by the hosting web server for a second original web page from a second content provider web server;

    parsing the second original web page for a first set of desired content elements;

20

    inserting the first set of desired content elements into a web page template containing a hosting web server format, thus creating the second recast web page; and  
    serving the second recast web page to the client browser; wherein the appearance of both the first and the second recast pages when presented by the client browser is as though all elements originated at the hosting web server.

13. The method as recited in claim 1, further comprising the steps of:

    determining client specific information about the client browser from which the request originated;

    selecting among a set of web page templates for the hosting server based on the client specific information, wherein each of the web page templates contains a different respective format; and

    using the selected web page template for creating the recast web page.

14. The method as recited in claim 13, wherein a plurality of web sites are serviced by the hosting web server and respective ones of the web page templates are used for each of the web sites.

15. A system for recasting web content on a hosting site, comprising:

    means for generating a request by the hosting web server for an original web page from a content provider web server;

    means for parsing the original web page for a first set of desired content elements;

    means for inserting the first set of desired content elements into a web page template containing a hosting web server format, thus creating a recast web page; and  
    means for serving the recast web page to a client browser; wherein the appearance of the recast page when presented by the client browser is as though all elements originated at the hosting web server.

16. The system as recited in claim 15, wherein one of the desired content elements is an advertisement element from the content provider web server, and the system further comprises means for inserting a call back to the content provider web server for the advertising element.

17. The system as recited in claim 15 further comprising: a cache for caching the desired content from the original page at the hosting web server; and

    means for determining whether there is a more recent version of the original page at the content provider server;

    wherein the cached desired content is used if there is no more recent version of the original page to respond to the second request.

18. The system as recited in claim 17 further comprising: a store of URLs to content provider servers for cached content in the cache; and

    means for periodically polling the URLs to determine whether there is a more recent version of any of the cached content at any of the stored URLs.

19. The system as recited in claim 17 further comprising: means for parsing the most recent version of the original web page for the first set of desired content elements; means for inserting the first set of desired content elements into a web page template containing a hosting web server format, thus creating a new version of the recast web page; and

    means for serving the new version of the recast web page to the client browser responsive to the second request;

6,128,655

**21**

wherein the desired content elements from the more recent version of the original page are cached at the hosting server.

**20.** The system as recited in claim **15** further comprising:  
a store of filter definitions;

means for searching for a filter definition for web pages from the content provider server in the store of filter definitions.

**21.** The system as recited in claim **20** wherein the store of filter definitions includes a default filter to be used when no filter definition exists for the content provider server.

**22.** The system as recited in claim **15**, wherein at least some of the content in the recast page originates at a second content provider server and when the recast page is presented by a client browser, both the first and second set of desired content elements look native to the hosting server.

**23.** The system as recited in claim **15**, wherein the web page template contains navigational features for the hosting server.

**24.** The system as recited in claim **15**, further comprising means for processing the desired content elements to eliminate harmful code, prior to insertion in the web page template.

**25.** The system as recited in claim **15**, further comprising means for revising relative links in the original page to links appropriate to the recast web page.

**26.** The system as recited in claim **15**, further comprising:  
means for determining client specific information about the client browser from which the request originated; and

means for selecting among a set of web page templates for the hosting server based on the client specific information, wherein each of the web page templates contains a different respective format;

wherein the selected web page template is used for creating the recast web page.

**27.** The system as recited in claim **26**, wherein a plurality of web sites are serviced by the hosting web server and respective ones of the web page templates are used for each of the web sites.

**28.** A computer program product for recasting web content on a hosting site, comprising:

means for generating a request by the hosting web server for an original web page from a content provider web server;

means for parsing the original web page for a first set of desired content elements;

means for inserting the first set of desired content elements into a web page template containing a hosting web server format, thus creating a recast web page; and

means for serving the recast web page to a client browser; wherein the appearance of the recast page when presented by the client browser is as though all elements originated at the hosting web server.

**22**

**29.** The product as recited in claim **28**, wherein one of the desired content elements is an advertisement element from the content provider web server, and the system further comprises means for inserting a call back to the content provider web server for the advertising element.

**30.** The product as recited in claim **28** further comprising:  
means for caching the desired content from the original page at the hosting web server; and

means for determining whether there is a more recent version of the original page at the content provider server;

wherein the cached desired content is used if there is no more recent version of the original page to respond to the second request.

**31.** The product as recited in claim **28** further comprising:  
a store of URLs to content provider servers for cached content in the cache; and

means for periodically polling the URLs to determine whether there is a more recent version of any of the cached content at any of the stored URLs.

**32.** The product as recited in claim **28** further comprising:  
means for storing a set of filter definitions; and  
means for searching for a filter definition for web pages from the content provider server in the set of filter definitions.

**33.** The product as recited in claim **28**, wherein the web page template contains navigational features for the hosting server.

**34.** The product as recited in claim **28**, further comprising means for processing the desired content elements to eliminate harmful code, prior to insertion in the web page template.

**35.** The product as recited in claim **28**, further comprising means for revising relative links in the original page to links appropriate to the recast web page.

**36.** The product as recited in claim **28**, further comprising:  
means for determining client specific information about the client browser from which the request originated; and

means for selecting among a set of web page templates for the hosting server based on the client specific information, wherein each of the web page templates contains a different respective format;

wherein the selected web page template is used for creating the recast web page.

**37.** The product as recited in claim **36**, further comprising means for servicing a plurality of web sites by the hosting web server and respective ones of the web page templates are used for each of the web sites.

\* \* \* \* \*



(12) **United States Patent**  
**Harter et al.**

(10) **Patent No.:** **US 6,212,564 B1**  
(45) **Date of Patent:** **Apr. 3, 2001**

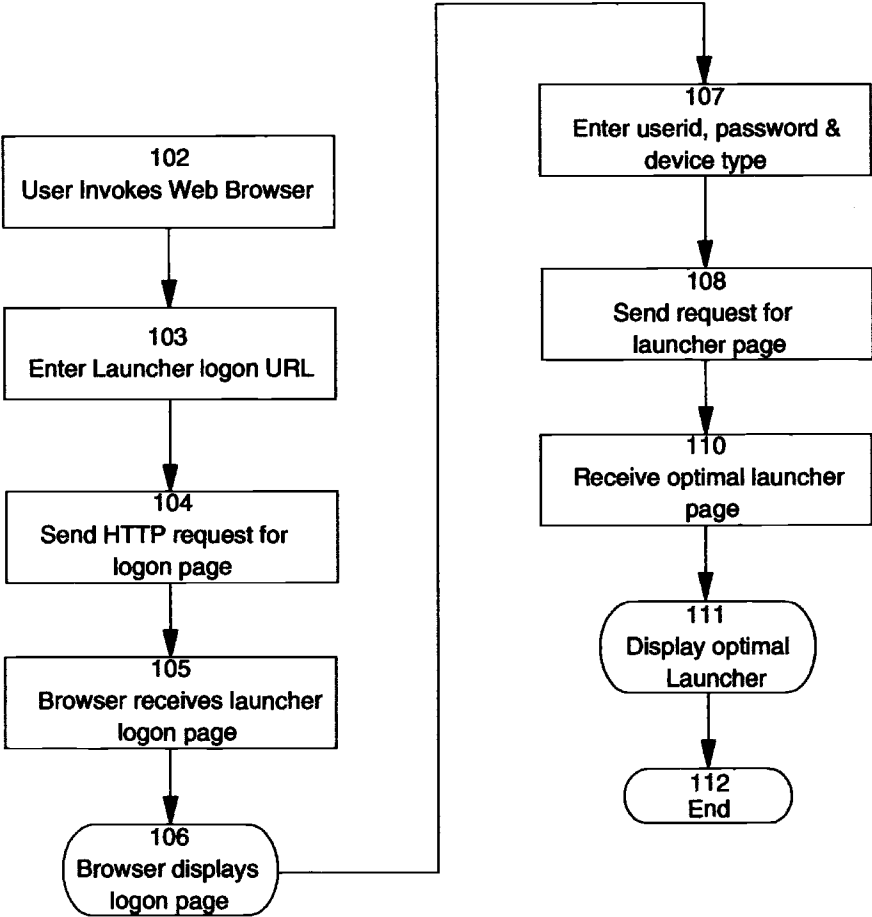
- (54) **DISTRIBUTED APPLICATION LAUNCHER FOR OPTIMIZING DESKTOPS BASED ON CLIENT CHARACTERISTICS INFORMATION**
- (75) Inventors: **John L. Harter**, Cary; **David Bruce Llection**, Raleigh, both of NC (US)
- (73) Assignee: **International Business Machines Corporation**, Armonk, NY (US)
- (\*) Notice: Subject to any disclaimer, the term of this patent is extended or adjusted under 35 U.S.C. 154(b) by 0 days.
- (21) Appl. No.: **09/108,813**
- (22) Filed: **Jul. 1, 1998**
- (51) **Int. Cl.**<sup>7</sup> ..... **G06F 17/30**
- (52) **U.S. Cl.** ..... **709/228; 709/227; 709/246; 707/526; 707/546**
- (58) **Field of Search** ..... **707/103, 526, 707/542; 709/246, 227, 228; 713/200**

- (56) **References Cited**  
**U.S. PATENT DOCUMENTS**
- |           |   |         |                |         |
|-----------|---|---------|----------------|---------|
| 5,907,324 | * | 5/1999  | Larson et al.  | 345/330 |
| 5,908,469 | * | 6/1999  | Botz et al.    | 713/201 |
| 5,928,323 | * | 7/1999  | Gosling et al. | 709/203 |
| 5,983,267 | * | 11/1999 | Shklar et al.  | 709/217 |
| 6,012,098 | * | 1/2000  | Bayeh et al.   | 709/246 |
| 6,044,465 | * | 3/2000  | Dutcher        | 713/200 |
- \* cited by examiner
- Primary Examiner*—Thomas G. Black  
*Assistant Examiner*—William Trinh  
(74) *Attorney, Agent, or Firm*—Jeanine S. Ray-Yarletts

(57) **ABSTRACT**

A method, system and program product are described which allow a dynamically constructed desktop to be optimized for the capabilities of the browser in the target device. This enables a server to manage the distribution of applets to clients matching the capabilities of the browser, the Java runtime environment (JRE), the screen resolution, color depth, sound capabilities, link speed, communications links and any other items which can be differentiated that are supported by the client requesting use of the applet.

**7 Claims, 6 Drawing Sheets**



U.S. Patent

Apr. 3, 2001

Sheet 1 of 6

US 6,212,564 B1

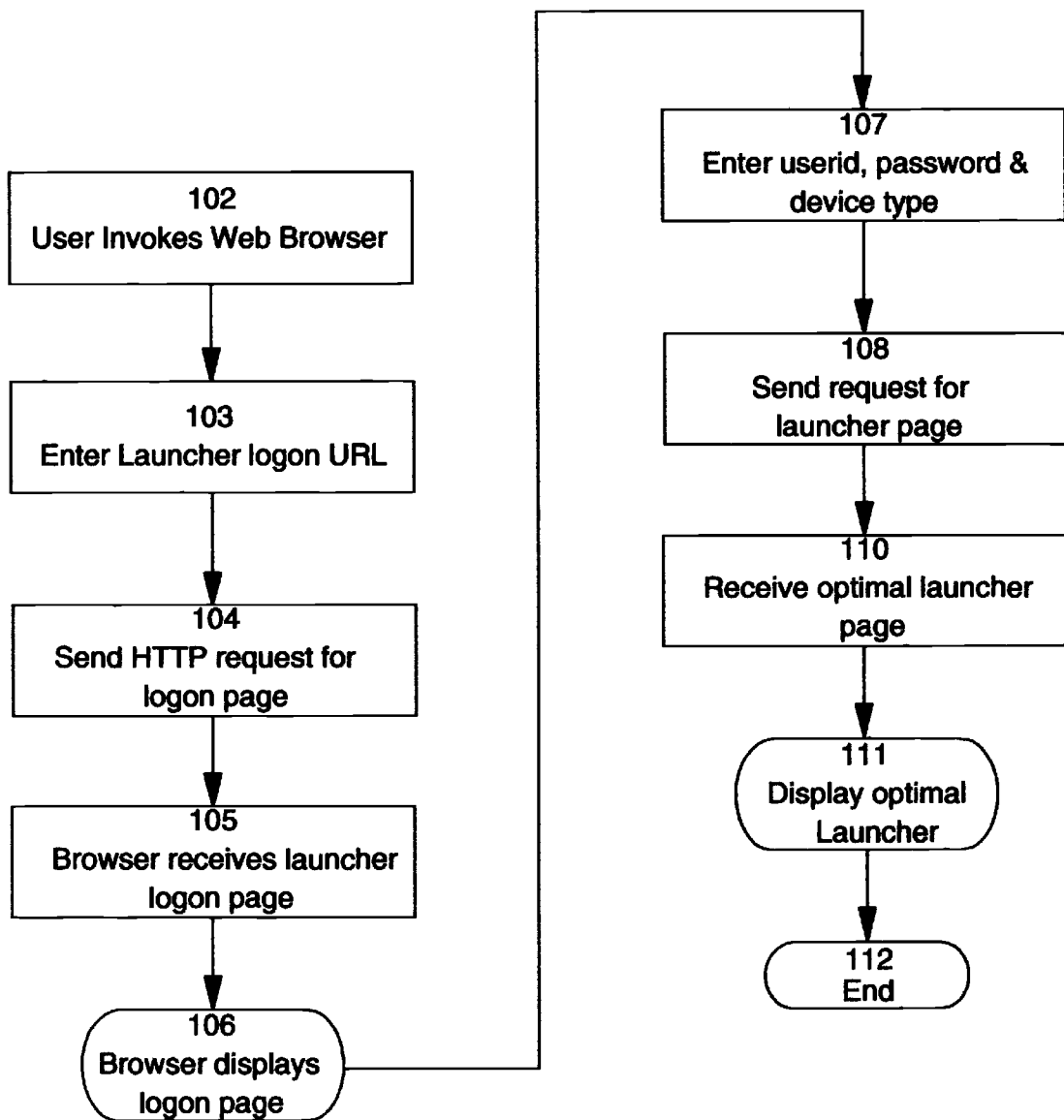


FIG. 1